Group Location Selection Queries over Uncertain Objects

Chuanfei Xu, Yu Gu, Roger Zimmermann, *Member, IEEE*, Shukuan Lin, Ge Yu, *Member, IEEE*

Abstract— Given a set of spatial objects, facilities can influence the objects located within their influence regions which are represented by circular disks with the same radius r. Our task is to select the minimum number of locations such that establishing a temporary facility at each selected location would ensure that all the objects are influenced. Aiming to solve this location selection problem, we propose a novel kind of location selection query, called *group location selection* (*GLS*) queries. In many real-world applications, every object is usually located within an *uncertainty region* instead of at an exact point. Due to the uncertainty of the data, *GLS* processing needs to ensure that the probability of each uncertain object being influenced by one facility is not less than a given threshold τ. An analysis of the time cost reveals that it is infeasible to exactly answer *GLS* queries over uncertain objects in polynomial time. Hence, this paper proposes an approximate query framework for answering queries efficiently while guaranteeing that the results of *GLS* queries are correct with a bounded probability. The performance of the proposed methods of the framework is demonstrated by theoretical analysis and extensive experiments with both real and synthetic datasets.

Index Terms—Group location selection, uncertain object, sampling method, coverage set

1 Introduction

Recently, a set of related problems concerning location selection has gained significant research attention for spatial decision support systems. A number of studies [1]–[6] have proposed solutions to various instances of such problems. In this paper we consider a novel location selection problem termed *group location selection* which cannot be solved by existing techniques. This problem is motivated by the following examples.

Considering a military application example, we wish to destroy all targets in a battlefield rapidly (e.g., enemy emplacements). To achieve this task, we need to efficiently select a group of locations at which bombs will be dropped to destroy these targets. Theoretically, each bomb is able to destroy things whose distances to the explosion point are not more than a fixed value, and thus any target whose distance to the explosion point is more than this fixed value cannot be destroyed by this bomb. As shown in Figure 1(a), when a bomb is dropped at a bomb-dropping location p_j, the targets o_1, o_2 and o_3 located within its influence region can be destroyed. The influence region is denoted by a circular disk with the center at p_j. Note that one bomb is enough for destroying the targets in its influence region, so our objective is to drop only one bomb at each selected location. Due to the expensive cost of bombs, it is desirable to destroy all the targets with the minimal number of bombs. This goal motivates a query for selecting the minimal number of bomb-dropping locations such that dropping a bomb at each selected location will ensure that all the targets can be destroyed.

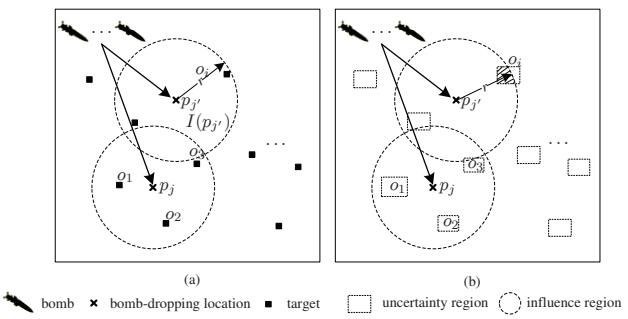


Fig. 1. Motivating example

As described in the above scenario, we desire that the bomb-dropping locations can be returned efficiently because a real-time response is crucial in any battle. In Figure 1(a), objects (i.e., targets) o_1, o_2 and o_3 can be influenced (destroyed) by exploding a bomb at location p_j. We regard a bomb explosion as a special case of setting up a temporary facility which has an influence region. For ease of presentation, we assume that each influence region is a circular disk with a fixed radius. For instance, the influence region $I(p_{j'})$ is a circular disk with its center at the location $p_{j'}$ and a radius r. In the two-dimensional Euclidean space $\mathcal{U}$, the object o_i can be influenced by a facility located at the location $p_{j'}$ if and only if

$$dist(o_i, p_{j'}) \leq r. \quad (1)$$

- *Chuanfei Xu, Yu Gu, Shukuan Lin, Ge Yu are with the Dept. of Information Science & Engineering, Northeastern University, China. E-mail: xuchuanfei@research.neu.edu.cn, {guyu,linshukuan,yuge}@ise.neu.edu.cn.*
- *Roger Zimmermann is with the Dept. of Computing Department of & Computer Science, National University of Singapore, Singapore. E-mail:rogerz@comp.nus.edu.sg.*

Digital Object Indentifier 10.1109/TKDE.2012.160

In the described application, each object needs to be influenced by at least one temporary facility.

Many other applications, such as profile-based managements [6], face the common problem of "how to efficiently select a minimal number of locations at which all the given objects can be influenced by temporary facilities". However, existing location selection techniques [3]–[5] choose only one location to influence the maximum number of objects, and hence are not suitable for the above problem of group location selection.

Aiming to solve this problem, we propose a novel type of group location selection (*GLS*) query whose objectives are: (i) minimizing the number of selected locations, and (ii) ensuring that all objects can be influenced by the temporary facilities located at these locations. Due to measurement limitations [7] in positioning technologies (e.g., GPS), the locations of objects cannot be measured exactly in many cases. As described in [8], we assume that each object is located within a closed *uncertainty region*, which is denoted by a rectangle called *probabilistically constrained rectangle* (*PCR*). Moreover, each object's probability density function (PDF) is defined within the *PCR* region such that the integration of the PDF inside the rectangle is equal to one. Compared to deterministic objects that are represented by exact points in the space $\mathcal{U}$, it is much more difficult to evaluate whether each uncertain object is located within the influence region of a temporary facility. As illustrated in Figure 1(b), in the above example, the locations of objects (targets) are uncertain and represented by their *PCRs*. Given a probability threshold τ ($0 < \tau \leq 1$), the probability that an uncertain object o_i can be influenced by a facility located at $p_{j'}$ is at least threshold τ if and only if the probability that o_i is located in the influence region of $p_{j'}$ is not less than τ, which is denoted as

$$Pr(dist(o_i, p_{j'}) \leq r) \geq \tau, (0 < \tau \leq 1). \quad (2)$$

Clearly, the probability threshold τ reflects the lower bound of the likelihood that any object is influenced. As probability thresholds are widely useful for query over uncertain data [9], *GLS* queries adapt to different probability thresholds as well, which ensure the probability's lower bound of each uncertain object being influenced. To the best of our knowledge, this is the first work that aims to select multiple locations to establish temporary facilities while ensuring that the probability of each object being influenced is not less than τ (for deterministic objects, $\tau = 1$). In our queries, deterministic objects (i.e., when the location of each object is denoted by a precise point) can be regarded as a particular case of uncertain objects (i.e., the probability that the object is influenced equals zero or one). Thus, we focus on the general examples of *GLS* queries over uncertain objects in Example 1.

Example 1: Given a set of uncertain objects $\mathcal{O}$=$\{o_1, o_2, \ldots, o_9\}$, the *GLS* query aims to find the minimum number of locations $\mathcal{P}$=$\{p_1, p_2, \ldots, p_{Min}\}$ to set up temporary facilities (i.e., a temporary facility is set up at each location) while ensuring that the probability of any object $o_i \in \mathcal{O}$ being located in $I(p_j)$ ($p_j \in \mathcal{P}$) is not less than the threshold τ, where $I(p_j)$ denotes the influence region of a facility located at p_j.

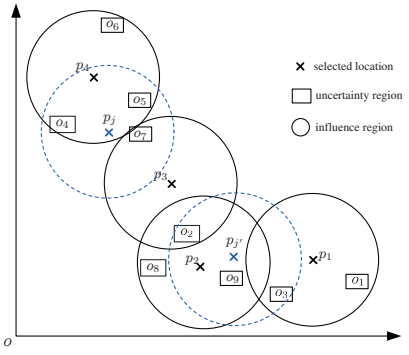


Fig. 2. An example of the *GLS* query

In the example of Figure 2, to influence all the objects it is sufficient to select a group of four locations $\{p_1, p_2, p_3, p_4\}$ at which to set up temporary facilities, respectively. Given a probability threshold τ, the probability of any object o_i ($1 \leq i \leq 9$) being influenced by temporary facilities at the selected locations is not less than τ. Also, there exist many groups of locations to set up temporary facilities which can influence these objects. For instance, we can also select $\{p_1, p_2, p_j, p_{j'}, p_4\}$ to influence each object with the probability not less than τ. Thus, an infinite number of groups of locations can be selected. Because we cannot verify whether the group of locations $\{p_1, p_2, p_3, p_4\}$ has a minimal number of locations for the *GLS* query in polynomial time with respect to the size of the given uncertain object set, according to the description in [10], the optimization problem for selecting the minimum number of selected locations is NP-Hard. Hence, the query evaluation raises serious practicality concerns for the size of the object set.

Aiming to address the above challenge, we propose a general query framework for approximately answering *GLS* queries in polynomial time while guaranteeing that the results of the queries are correct with a bounded probability. The contributions of our work can be summarized as follows.

- We formally define a novel type of location selection query, called group location selection (*GLS*) queries, which select the minimum number of locations such that setting up a temporary facility at each selected location will ensure that all the given objects are influenced.
- We propose a general framework for answering *GLS* queries over uncertain objects. In the framework, queries adapt to different probability thresholds which ensure a lower bound of probability such that each uncertain object is influenced.

- As part of the framework, we present algorithms for answering *GLS* queries in polynomial time while ensuring that the results of the queries are correct with a bounded probability by setting the proper number of samples.
- We evaluate the performance of these algorithms through theoretical analysis and extensive experiments with real and synthetic datasets.

The rest of the paper is organized as follows. Section 2 discusses the related work. Section 3 formalizes the *GLS* query and describes the query framework. Sections 4 and 5 provide effective methods for answering our queries. Section 6 presents the experimental results and Section 7 concludes the paper.

2 RELATED WORK

2.1 Location Selection Problem

The location selection problem is of significant importance in spatial query research. There exist a large volume of previous approaches [2] [5] [7] [11] [12] devoted to spatial query processing over uncertain objects. Wong et al. [5] address an RNN location selection problem called Bichromatic reverse nearest neighbors (MaxBRNN) which focuses on finding an optimal region that maximizes the size of BRNNs. An efficient algorithm called Maxoverlap is proposed as a solution. In [5], the authors present an efficient algorithm for the MaxBRNN problem where the optimization criteria is maximizing the number of potential objects for a facility in the Euclidean space. Considering that any object may use the facility which is any of his/her k nearest facilities, the work in [6] extends the MaXBRNN problem to MaxBRkNN which finds an optimal region with the property that deploying a facility in this region attracts the maximum number of customers. Moreover, Max-influence problems maximize the influence of a facility, where the influence is typically defined as the number of objects that are the RNNs of this facility, which are a significant type of location selection queries. Cabello et al. [2] propose the MAXCOV problem. They introduce the nearest location circle (NLC) as part of their solution. The MAXCOV problem is to find the regions that are enclosed by the largest number of NLCs. Because these studies do not consider the problem of selecting multi-locations to set up facilities, their solutions are not suitable for our problem.

In addition, several query approaches have been proposed for retrieving the optimal locations with respect to different semantics. Du et al. [4] define and investigate a location selection problem. They focus on solving the problem in the Manhattan distance space. They retrieve objects of interest in some given order and then use a plane-sweep algorithm to identify the best location. Furthermore, Xia et al. [13] examine a related problem of finding the top-t most influential facilities among a given set of facilities. Note that in the top-t influential facilities problem, the influence of a facility is defined as the total weight of its RNNs. Qi et al. [14]'s method defines the mindist optimization function which is a secondary optimization for the location selection problem. For the above queries, however, considerable improvements are required to extend them to multiple facilities that influence objects. The solutions for location selection queries cannot be applied to find a group of locations for multiple facilities and these methods do not consider the locations of objects to be uncertain.

2.2 Other Related Problem

The spatial assignment problem also has attracted much research attention [15], [16] in the optimization problem domain. This optimization problem desires an optimal assignment between objects and a set of given facilities. U et al. [15] consider a capacity constrained assignment, where each facility can influence at most k objects, the total number of objects is maximized and the total assignment cost is minimized. Furthermore, U et al. [16] also propose the continuous optimal assignment problem, whose objective is to construct an optimal assignment between mobile objects and a set of given facilities. To solve this problem, the work first accelerates the initial assignment computation by exploiting geometric properties and splitting the problem into smaller sub-problems, and then solving those using an off-the-shelf optimal assignment algorithm. However, the spatial assignment problem does not provide a minimal number of locations, and thus it does not directly translate to the group location selection problem proposed by us. In addition, the set covering problem [17], [18] was proposed to minimize the cost of sets which can cover all data in a given set, where each datum in the set is a 1-dimensional value. However, this problem does not need to consider the space relations between objects and selected locations so that the solution for the problem is not suitable for *GLS* queries.

2.3 Summary of Location Selection Methods

Table 1 summarizes the differences between our problem and previously studied location selection problems, where $\mathcal{F}$ represents the set of facilities which have been established at some fixed locations, and $\mathcal{O}$ or $\mathcal{P}$ denote an object set or a facility set, respectively. Furthermore, our problem does not belong to the class of uncertain data clustering problems [19]. These problems only consider the distances among objects in a cluster so that they are unable to ensure that all the objects are located in the influence regions of facilities. To the best of our knowledge, our approach is the first that is able to return a minimal number of locations at which facilities are set up to influence all the objects.

TABLE 1
Location selection problems and solutions

Problem	Results	Solution space	Distance function	Dataset
MaxBRNN [5]	A region	Continuous	L_2	$\mathcal{O}$, $\mathcal{F}$
MAXCOV [2]	A region	Continuous	L_2	$\mathcal{O}$
Min-dist optimization [14]	A location	Discrete	L_2	$\mathcal{O}$, $\mathcal{P}$, $\mathcal{F}$
Spatial assignment optimization [15]	Assignment relations	Discrete	L_2	$\mathcal{O}$, $\mathcal{P}$
Proposed	A group of locations	Continuous	L_2	$\mathcal{O}$

3 PRELIMINARIES

In this section, we first formally define the *GLS* query processing. Then we devise a general framework for answering these queries. Table 2 summarizes the symbols used in this paper.

3.1 Problem Definition

We use $\mathcal{O} = \{o_1, \ldots, o_i, \ldots, o_n\}$ to denote a set of uncertain objects. In this work, any object $o_i \in \mathcal{O}$ is located within its own *probabilistically constrained rectangle* (*PCR*). In the rectangular region, the PDF of each object is defined, and the integration of this PDF inside the *PCR* is equal to one [9]. Moreover, we use the Euclidean distance as the measurement distance between any two locations in the two-dimensional Euclidean space $\mathcal{U}$. For any location p_j in the space $\mathcal{U}$, the influence region $I(p_j)$ is a circular disk with the center at p_j and the common radius r.

TABLE 2
Commonly used symbols

Symbol	Description
$\mathcal{O}$	a set of uncertain objects
$\mathcal{P}$	a set of selected locations
$\mathcal{C}_k$	coverage set
n	cardinality of $\mathcal{O}$
T	number of coverage sets
o_i	an uncertain object in $\mathcal{O}$
PCR_i	*PCR* of o_i
r	radius of the influence regions
$dist()$	distance function
τ	probability threshold

In order to select locations to establish temporary facilities, we need to know which objects can be influenced by one facility. We define a *coverage set* as a type of object sets in which the probability that each object can be influenced by one temporary facility is not less than the threshold τ.

Definition 1: (Coverage Set) Given a set of uncertain objects $\mathcal{O}$ and a probability threshold τ, a coverage set $\mathcal{C}_k$ is defined as a subset of $\mathcal{O}$, which satisfies (i) $\exists p_j \bigwedge Pr(dist(o_i, p_j) \leq r) \geq \tau$ for $\forall o_i \in \mathcal{C}_k$; and (ii) $\neg\exists p_{j'}(p_{j'} \neq p_j) \bigwedge Pr(dist(o_i, p_{j'}) \leq r) \geq \tau$ for $\forall o_i \in \mathcal{C}' \bigwedge \mathcal{C}_k \subsetneqq \mathcal{C}_{k'}$, where $\mathcal{C}_{k'}$ denotes any other coverage set.

From Definition 1, any coverage set cannot be a subset of any other coverage set. In a coverage set, the probability that each uncertain object can be influenced by one temporary facility is not less than the threshold τ. In Figure 2, for instance, the *PCRs* of o_1 and o_3 are completely covered by the influence region $I(p_1)$ so that the probabilities of o_1 and o_3 being influenced by a temporary facility located at p_1 is one. Since there is no other location to set up a temporary facility that can influence o_1 and o_3, $\{o_1, o_3\}$ is a coverage set. In this example, we are able to find many other coverage sets such as $\{o_2, o_8, o_9\}$ and so on. For any coverage set, the probability that any object contained in this coverage set can be influenced by one temporary facility is not less than τ. To influence all the uncertain objects, we need to choose several coverage sets which together contain all the objects in $\mathcal{O}$. *GLS* queries wish to minimize the number of selected locations, which is related to the number of chosen coverage sets. We summarize the relation in the following lemma.

Lemma 1: Let $\bigcup^{k\in K} \mathcal{C}_k$ be union sets of $|K|$ coverage sets and $\bigcup^{k\in K} \mathcal{C}_k = \mathcal{O}$. If $|K|$ is the minimum number of coverage sets in order to contain all the uncertain objects in $\mathcal{O}$, then at least $|K|$ locations must be selected to set up temporary facilities which can influence all the objects in $\mathcal{O}$.

Proof: Suppose that we can select $|K'|$ ($|K'| < |K|$) locations (at each location one temporary facility is established) for establishing the temporary facilities to influence each uncertain object in $\mathcal{O}$ with the probability that is not less than the given probability threshold τ. According to the definition of coverage sets, some objects that can be influenced by one facility make up the subset of a coverage set. Therefore, there exist $|K'|$ coverage sets which include all the uncertain objects in $\mathcal{O}$. Since $|K|$ is the minimum number of coverage sets which include all the uncertain objects in $\mathcal{O}$, it is a contradiction to the assertion that $|K'| < |K|$. Hence, our assumption is not correct. That is to say $|K'| \geq |K|$. It follows that $|K|$ is the minimum number of selected locations to set up temporary facilities which can influence all the objects in $\mathcal{O}$. □

As described in the above lemma, we can minimize the number of selected locations by finding the minimum number of coverage sets which contain all the uncertain objects in $\mathcal{O}$. To find the minimum number of coverage sets, this work focuses on two main concerns: (i) how to retrieve all the coverage sets according to a given probability threshold τ (solved in Section 4) and (ii) how to choose the minimum number of coverage sets to contain all the objects in

$\mathcal{O}$ (solved in Section 5). For any coverage set $\mathcal{C}_k$, we need to retrieve a location p_j such that setting up a temporary facility at p_j would influence each object in $\mathcal{C}_k$ with the probability that is not less than the threshold τ. Such locations (e.g., p_j) are crucial for our solution. Then we define this type of location formally as follows.

Definition 2: (Feasible Location) Given a coverage set $\mathcal{C}_k$ and a probability threshold τ, *feasible locations* are defined as locations such that setting up a temporary facility at one of these locations p_j ensures $Pr(dist(o_i, p_j) \leq r) \geq \tau$ for $\forall o_i \in \mathcal{C}_k$.

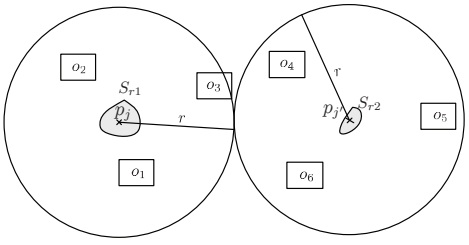


Fig. 3. Feasible locations for coverage sets

From this definition, a temporary facility established at any feasible location for a coverage set can influence every object in this coverage set with the probability not less than the given threshold τ. As shown in Figure 3, $\{o_1, o_2, o_3\}$ and $\{o_4, o_5, o_6\}$ are two coverage sets in which o_1, o_2, o_3 and o_4, o_5, o_6 can be influenced by one temporary facility respectively. We assume that all the feasible locations for the two coverage sets make up the regions S_{r1} or S_{r2}, respectively. Figure 3 illustrates that establishing a facility at any feasible location p_j in S_{r1} would guarantee that o_1, o_2, and o_3 are covered by the influence region $I(p_j)$ with a probability not less than τ. Likewise, a facility set up at a feasible location $p_{j'}$ in S_2 ensures that the probabilities of o_4, o_5, o_6 being covered by $I(p_{j'})$ are not less than τ. In most cases, we are unable to find the regions S_{r1} and S_{r2} exactly. Moreover, a facility that is set up at different feasible locations in S_{r1} (or S_{r2}) influences the same objects o_1, o_2, o_3 (or o_4, o_5, o_6), so that we solely expect to find two feasible locations within S_{r1} (and S_{r2}), respectively. The method of finding the feasible locations will be presented in the next sections.

Now we formally define group location selection (*GLS*) queries over uncertain objects, which are able to retrieve the minimum number of feasible locations such that establishing a temporary facility at each feasible location will ensure the probability of each object being influenced is not less than τ.

Definition 3: (Group Location Query over Uncertain Objects) Given a set of uncertain objects $\mathcal{O}$, a group location selection query, $GLS(\mathcal{O}, r, \tau)$, retrieves a group of feasible locations to set up the minimum number of temporary facilities for the coverage sets which include all the uncertain objects in $\mathcal{O}$. It is formally defined as

$$\begin{aligned} GLS(\mathcal{O}, r, \tau) \quad &= \{< p_j, \mathcal{C}_k > | \forall o_i \in \mathcal{C}_k, \\ &(i) Pr(dist(o_i, p_j) \leq r) \geq \tau; \\ &(ii) \bigcup_{k \in K} \mathcal{C}_k = \mathcal{O}; \\ &(iii) \neg \exists K', |K'| < |K|, \bigcup_{k' \in K'} \mathcal{C}_{k'} = \mathcal{O}\}, \end{aligned} \tag{3}$$

where $K \subseteq \{1, 2, \ldots, T\}$, $K' \subseteq \{1, 2, \ldots, T\}$, and r denotes the radius of the influence region of p_j.

As mentioned in the above definition, a group of locations whose number is minimized is retrieved by the *GLS* query. For each result coverage set $\mathcal{C}_k$, the *GLS* query needs to find at least one feasible location p_j. We next describe the framework for the *GLS* query processing.

3.2 The Query Framework

To answer *GLS* queries over uncertain objects efficiently, we now devise a general query framework.

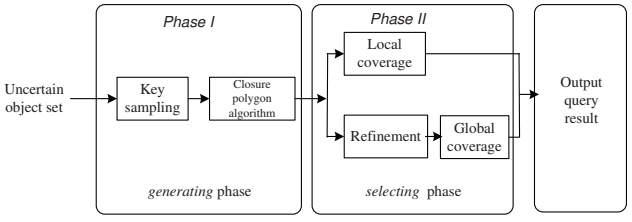


Fig. 4. The query framework for *GLS* queries

As illustrated in Figure 4, the query framework is composed of two phases: the *generating* phase and the *selecting* phase. In the generating phase, the framework needs to retrieve feasible locations for all coverage sets. Due to the uncertainty of objects' locations, it is necessary to evaluate the probability that each object in any coverage set is influenced by a facility located at a feasible location for this coverage set. Hence, the time cost of searching these feasible locations for all coverage sets becomes prohibitive if the size of the object set increases. To solve this problem, we first present a sampling method which is able to approximately find the feasible locations for all coverage sets. Furthermore, we evaluate the probability bound (i.e., the probability that any coverage set is accurate and equal to the bound) of our sampling method. The bound is adjustable by setting a proper number of samples. Based on the sampling method, the closure polygon algorithm is proposed to find the feasible locations for all coverage sets in this phase (see Section 4).

According to Lemma 1, to minimize the number of selected locations, we must find the lowest number of coverage sets from all the coverage sets generated in the first phase. In the selecting phase, a greedy location selection approach (termed local coverage algorithm) repeatedly chooses the coverage set that contains the largest number of objects. This approach returns the chosen coverage sets until all objects are

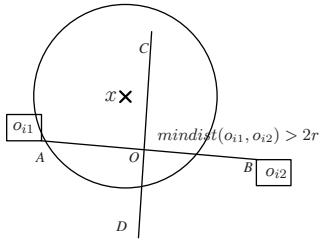


Fig. 5. Proof sketch of Lemma 2

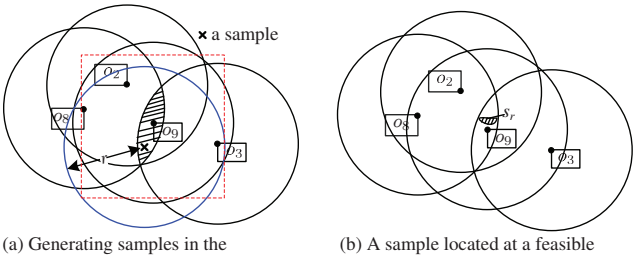
contained. However, this approach is unable to ensure that the number of the chosen coverage sets containing all objects is minimized. To improve the solution, another location selection approach (termed global coverage algorithm) is proposed to choose the minimal number of coverage sets. Compared to the local coverage algorithm, the global approach can choose a minimal number of coverage sets. To improve its efficiency, this approach refines intermediate results for *GLS* queries (see Section 5). Note that even though the approach of the selecting phase is exact, the sampling process is an approximate method so that our framework for *GLS* queries is also approximate. Along with the increasing number of samples, the framework becomes more precise with a higher likelihood, which is described in Section 5.3.

4 GENERATING COVERAGE SETS

In the generating phase, we need to verify which objects can be in a coverage set. We assume that each uncertain object is represented by a Boolean variable (i.e., if an object can be in a coverage set, this Boolean variable is set to 1; otherwise, the variable is set to 0). Using these Boolean variables, we can construct a disjunctive normal form (DNF) to represent the case that several objects can be influenced by one temporary facility located at p_j and any additional object cannot be influenced by a temporary facility located at any other location with a probability not less than the given threshold τ. For example, there are four uncertain objects $\{o_1, o_2, o_3, o_4\}$. If $\{o_1, o_2, o_3\}$ and $\{o_1, o_4\}$ can be influenced by one temporary facility respectively, a DNF is denoted by $(x_{11} \wedge x_{21} \wedge x_{31} \wedge \neg x_{41}) \vee (x_{12} \wedge \neg x_{22} \wedge \neg x_{32} \wedge x_{42})$ (x_{ij} is the Boolean variable that denotes o_i in $\mathcal{C}_1$ or $\mathcal{C}_2$, $1 \leq i \leq 4$, $1 \leq j \leq 2$). When $(x_{11} \wedge x_{21} \wedge x_{31} \wedge \neg x_{41}) = 0$, $x_{41} = 1$. Therefore, in this case $\mathcal{C}_1$ should be $\{o_1, o_2, o_3, o_4\}$ so that $\{o_1, o_2, o_3\}$ is not a coverage set. Likewise, when $(x_{12} \wedge \neg x_{22} \wedge \neg x_{32} \wedge x_{42}) = 0$, $\{o_1, o_4\}$ is also not a coverage set. That is to say, if the DNF satisfies $(x_{11} \wedge x_{21} \wedge x_{31} \wedge \neg x_{41}) = 1$ and $(x_{12} \wedge \neg x_{22} \wedge \neg x_{32} \wedge x_{42}) = 1$, $\{o_1, o_2, o_3\}$ and $\{o_1, o_4\}$ are coverage sets. As described in [20], finding a solution to satisfy the above conditions lead to SAT. Hence, the problem of finding all coverage sets is NP-complete. In this section, we propose an approximate method to find all coverage sets efficiently.

4.1 Finding Feasible Locations

To improve the query efficiency, all the objects in $\mathcal{O}$ are indexed by an R-tree in the space $\mathcal{U}$. In some cases, some objects cannot be influenced by a single temporary facility, so that these objects cannot be members of the same coverage set. We can identify these cases as follows.



(a) Generating samples in the shadowed region

(b) A sample located at a feasible location

Fig. 6. Finding feasible locations

Lemma 2: Let o_{i1} and o_{i2} be any two uncertain objects and r be the radius of each temporary facility. If the minimum distance $min\_dist(o_{i1}, o_{i2}) > 2r$, then they cannot be influenced by one temporary facility.

Proof: As shown in Figure 5, $|AB|$ denotes the minimum distance between o_{i1} and o_{i2}, CD denotes the perpendicular bisector of AB and O denotes the midpoint of AB. Since $min\_dist(o_{i1}, o_{i2}) > 2r \Rightarrow |AB| > 2r \Rightarrow |AO| = |OB| > r$, $min\_dist(x, o_{i2}) > |OB| > r$ if any point x is located on the left of CD (or $min\_dist(x, o_{i1}) > |OA| > r$ if x is located on the right of CD). Therefore, no matter at which point a temporary facility is set up, o_{i1} or o_{i2} cannot be both influenced by the same temporary facility. □

In Example 1, there exists a likelihood that a set of objects $\{o_2, o_3, o_8, o_9\}$ is a coverage set, since the minimum distances between any two objects are not more than $2r$. Therefore, the circular disks with the centers at neighboring vertices of different objects' *PCRs* have an intersection region S (i.e., the shadowed region in Figure 6(a)). Since the distance between any point in the intersection region and the location of any object is not more than r, a temporary facility established in this shadowed region can influence the objects with non-zero probabilities. The probability of an object o_i being influenced by a facility located at p_j is denoted as

$$Pr(dist(o_i, p_j) \leq r) = \int_{p_j \in S \wedge u \in I(p_j) \cap PCR_i} PDF_{o_i} du, \quad (4)$$

where PDF_{o_i} represents the PDF of o_i located in its own PCR_i. Using the above equation, we can compute the probability that each object is influenced by a facility located at p_j in the shadowed region S. For a set of objects, if we can find a point p_j such that the probability of each object influenced by the temporary facility built at p_j is not less than τ, this set of objects is a coverage set. As mentioned previously, the problem

of finding all coverage sets is NP-complete so that the time cost of this process becomes prohibitive.

Aiming to address this challenge, a novel sampling method is proposed to select key samples. Each key sample represents a feasible location such that setting up a temporary facility at this feasible location will influence any object in a coverage set with a probability not less than τ. The sampling method is illustrated in Figure 6(a). We first find a regular shape (e.g., a rectangle) which contains the intersection of circular disks (i.e., the shadowed region in which we wish to sample). Then we uniformly generate samples in the regular shape. Because many samples are not in the shadowed region, we need to evaluate the distances between each sample and the centers of the circular disks. If the distances are not more than the radius of circular disks r, the sample must be in the shadowed region. The samples that are not within the shadowed region are deleted, and thus the remaining samples are within the shadowed region uniformly. If a facility is set up in the shadowed region, each object is influenced by this facility with a non-zero probability. Suppose that we generate samples outside the shadowed region. There is at least one object (e.g., o_3) whose distance to any sample is not less than r. So picking a sample outside the shadowed region would result in that o_3 is not in the coverage set obtained by our sampling method. To ensure that obtained coverage sets are accurate, the sampling method generates samples in the shadowed region.

For a sample s at which we set up a facility, a set of objects $\mathcal{C}$ is influenced. Given any other sample s', we can find any set of objects $\mathcal{C}'$ in which the probability of each object being influenced by a facility located at s' is not less than τ. If $\mathcal{C} \nsubseteq \mathcal{C}'$, the location of s is selected as a feasible location for $\mathcal{C}$. This sample s is defined as a key sample, and all the key samples are represented as a set,

$$\mathcal{O}_S = \left\{ s \mid \mathcal{C} \nsubseteq \mathcal{C}', \begin{array}{l} \mathcal{C} = \{o_i | Pr(dist(o_i, s) \leq r) \geq \tau\} \\ \mathcal{C}' = \{o_{i'} | Pr(dist(o_{i'}, s') \geq r) \leq \tau\} \end{array} \right\}. \quad (5)$$

As illustrated in Figure 6(b), we assume that the shadowed region S_r denotes a set of feasible locations for a coverage set $\{o_2, o_3, o_8, o_9\}$. According to the definition of feasible locations, the probabilities that o_2, o_3, o_8 and o_9 can be influenced by a facility located within S_r are not less than the given threshold τ. To find this coverage set, we need to select any location in S_r as a feasible location. However, the shape of S_r is irregular so that it is difficult to find the feasible location in it. Hence, our sampling method generates M samples which are distributed in the intersection region S (the sample region in Figure 6(a)) uniformly. Using Eq. (5), the key samples are returned. If the probability (evaluated by Eq. (4)) of o_3 being influenced by a facility located at any key sample is less than τ, we consider $\{o_2, o_8, o_9\}$ as a coverage set. Since there exist cases where no key sample falls within S_r, the generated coverage set may be empty. Next, we define the probability that at least one key sample is a feasible location through the following lemma.

Lemma 3: Let S be the sample region, Sr be a set of feasible locations for a coverage set and M be the number of samples. Then, the probability that at least one key sample falls in Sr equals

$$1 - (1 - \frac{Sr}{S})^M. \quad (6)$$

Proof: Assume that the sampling method generates M samples which follow a uniform distribution in the region S. For any sample, its probability of being located within Sr is $\frac{Sr}{S}$. Thus, the probability that all the samples are not located within Sr is $(1 - \frac{Sr}{S})^M$. Note that the probability that at least one key sample is a feasible location equals $1 - (1 - \frac{Sr}{S})^M$. □

As stated in the lemma above, the probability that the sampling method obtains feasible locations is expected to be more than a given bound when M is large enough. Using our sampling method, we can find feasible locations with a bounded probability by utilizing the proper number of samples. According to these feasible locations, we are able to find each coverage set in which objects can be influenced by a facility located at a selected feasible location. All coverage sets can then be represented as

$$\mathcal{O} = \bigcup_{k=1}^{T} \mathcal{C}_k, \quad (7)$$

where T denotes the total number of coverage sets. Using the sampling method, a naive solution evaluates which objects have a common intersection region of their circular disks with the center at neighboring vertices and a radius of r (e.g., $\{o_2, o_3, o_8, o_9\}$ in Example 1). Then it uses the sampling method to find the feasible locations for all coverage sets. However, it is highly time-consuming to implement this naive solution algorithm, because it needs to calculate a massive number of integral computations as shown in Eq. (4). For example, let us solely consider two objects o_2 and o_3 at first. The circular disks of o_2 and o_3 have a common intersection region (denoted as a sample region) so we require the evaluation of two integral computations for the two objects. When considering one more object o_8, we have to execute the sampling method in a new sample region for the three objects again. Note that we need to calculate some integral computations for each sampling. Consequently, the time cost will be prohibitive. We present a more efficient algorithm in the next subsection.

4.2 Closure Polygon Algorithm

With the proposed sampling method the naive algorithm provides a solution for retrieving every coverage set by finding feasible locations. However, the cost of the massive integral computations is so complex that this algorithm is inefficient. To increase efficiency, we propose a new algorithm, termed closure polygon algorithm, which calculates a lower number of integral computations.

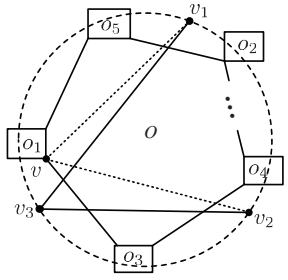


Fig. 7. Proof sketch of Lemma 4

Recall that before executing the sampling method, we need to find each sample region which is generated by some objects that can be influenced by a temporary facility with a certain probability. Hence, we wish to efficiently verify whether objects are covered by an influence region of one facility. We formalize this case as follows.

Lemma 4: Let $v_1, \ldots, v_m$ be m vertices of the closure polygon[1] of some given objects' *PCRs*, respectively. If $dist(v_i, v_{i'}) \leq \sqrt{3}r$, then there exists a region in which the influence region of a temporary facility intersects with all the *PCRs* of these objects.

Proof: As illustrated in Figure 7, uncertainty regions of some objects are represented by *PCRs*. The vertices $v_1, \ldots, v_m$ generate a polygon Pol which is the smallest closure polygon for the *PCRs*. Due to $dist(v_i, v_{i'}) \leq \sqrt{3}r$, suppose that v_1, v_2 and v_3 satisfy $dist(v_1, v_2) = \sqrt{3}r$, $dist(v_2, v_3) = \sqrt{3}r$ and $dist(v_3, v_1) = \sqrt{3}r$. O is a circle disk with v_1, v_2 and v_3 located on the boundary. For any other vertex v, it is obvious that $dist(v_2, v) \leq \sqrt{3}r$ and $dist(v, v_1) \leq \sqrt{3}r$. Thus, $\angle v_1 v v_2 \geq \angle v_1 v_3 v_2 = \frac{\pi}{3}$. It follows that v resides inside the circle disk O. Consequently the circle disk O can intersect with all the *PCRs* of these objects. Therefore the objects can be influenced by a temporary facility with a probability. □

In contrast to the above lemma, we also consider the case where some objects have no likelihood to be influenced by one facility.

Lemma 5: Given any probability threshold, the objects (at least three objects) whose pair-wise minimal distances are larger than $\sqrt{3}r$ cannot be influenced by only one temporary facility.

Proof: Assume that the minimal distance between any two objects in a set of uncertain objects is represented by $min\_dist(o_{i1}, o_{i2})$. For vertices of the closure polygon of these objects, $min\_dist(o_{i1}, o_{i2}) > \sqrt{3}r \Longrightarrow dist(v_i, v_{i'}) > \sqrt{3}r$. Analogous to Lemma 4, $\angle v_i v v_{i'} \geq \angle v_i v' v_{i'}$, where v is any point in the circle and v' is any point within a *PCR*. Thus, v' is located outside the circle. Note that these objects cannot be influenced by one temporary facility with any probability threshold. □

Using the above lemmas, we can use the distance relations among uncertain objects to decide whether they can be influenced by one temporary facility instead of complex integral computations. Based on the lemmas, we propose the closure polygon algorithm to find feasible locations for all coverage sets. The closure polygon algorithm is illustrated in Algorithm 1.

Algorithm 1: Closure Polygon Algorithm

```
   Input  : GLS(O, r, τ)
   Output: {< p_j, C_k > | O = ⋃_{k=1}^{T} C_k}
1  k ⟵ 1 and X ⟵ Φ
2  take any object from O into X
3  for each object o_i in O do
4      for each object o_j in X do
5          if min-dist(o_i, o_j) ≤ √3r then
6              take o_i into X
7              if all the objects in X satisfy Lemma 4
               then
8                  find each key sample s
9                  while Pr(dist(o_i, s) ≤ r) ≥ τ do
10                     take o_i into C_k
11     O' ⟵ O − C_k; k ⟵ k + 1
12     take any object from O' into X
13 T ⟵ k and k ⟵ 1
14 while k ≤ T do
15     select s as a feasible location p_j for C_k
16     return < p_j, C_k >
17     k ⟵ k + 1
```

The closure polygon algorithm first finds the objects that are influenced by one temporary facility (lines 2–7). Next it compares whether each object is influenced above a threshold probability τ (lines 8–10). Finally, the algorithm returns feasible locations for all the coverage sets (lines 13–17).

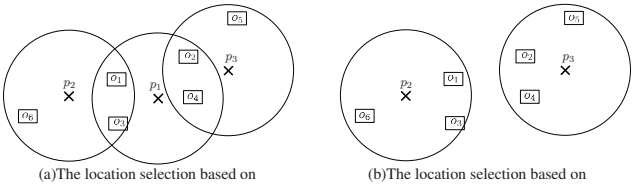
5 SELECTING COVERAGE SETS

5.1 Location Selection Based on Local Coverage

We use an R-tree to index the set of uncertain objects $\mathcal{O}$ in the two-dimensional space $\mathcal{U}$. As shown in Algorithm 1, all the coverage sets and the relevant feasible locations are returned. To answer *GLS* queries, we need a location selection method which chooses some feasible locations while guaranteeing that all uncertain objects are contained by coverage sets of these feasible locations. A local coverage algorithm is

1. This is a largest polygon which has no intersection region with the *PCRs* of these uncertain objects.

proposed to solve this problem in the second phase of our query framework. The key idea for this approach is to ensure that a coverage set containing the largest number of uncertain objects is preferentially selected, and hence the solution is a greedy method.



(a)The location selection based on local coverage

(b)The location selection based on global coverage

Fig. 8. Different coverage methods

The local coverage algorithm represents a simple way to answer the *GLS* query over uncertain objects. Nevertheless, while the algorithm obtains the local solution, it cannot ensure that the minimal number of locations is selected. We illustrate the algorithm and its drawback with an example. Given a set of objects $\mathcal{O} = \{o_1, o_2, o_3, o_4, o_5, o_6\}$, the feasible locations p_1, p_2, and p_3 for three coverage sets are shown in Figure 8(a), respectively. The coverage sets are denoted by $\mathcal{C}_1 = \{o_1, o_2, o_3, o_4\}$, $\mathcal{C}_2 = \{o_1, o_3, o_6\}$, and $\mathcal{C}_3 = \{o_2, o_4, o_5\}$. The local coverage algorithm first chooses $\mathcal{C}_1$ because $\mathcal{C}_1$ contains the largest number of objects. Since o_5 and o_6 also need to be contained by some coverage sets, $\mathcal{C}_2$ and $\mathcal{C}_3$ are also chosen by the local coverage algorithm. Therefore the three feasible locations p_1, p_2, and p_3 are returned. That is to say, the *GLS* query selects three locations at which three facilities are set up to ensure that the probability of each object being influenced is not less than the threshold τ. However, Figure 8(b) illustrates a better result which solely includes two locations p_2 and p_3, and two coverage sets $\mathcal{C}_2$ and $\mathcal{C}_3$. This illustrates that the local coverage algorithm may not return the optimal number of locations.

5.2 Location Selection Based on Global Coverage

We introduce a *global coverage algorithm* to choose coverage sets including all the given uncertain objects from all coverage sets. Since the coverage sets are generated by a sampling process, the coverage sets may be approximate. The global coverage algorithm can solely ensure that the minimal number of coverage sets is chosen given all the coverage sets. Using the proposed sampling method and this global algorithm to answer *GLS* queries, we will analyze the error margins of the results in Section 5.3.

Compared with the local coverage algorithm, the global coverage algorithm considers the global optimum so that the number of chosen coverage sets is minimized. Before introducing the selection process, we present refinement methods for improving the efficiency of *GLS* queries over uncertain objects. Recall the set of uncertain objects $\{o_1, o_2, o_3, o_4, o_5, o_6, o_7, o_8, o_9\}$ from Example 1, as illustrated in Figure 9. Assume we found all the coverage sets as follows: $\mathcal{C}_1 = \{o_2, o_8, o_9\}$, $\mathcal{C}_2 = \{o_1, o_3\}$, $\mathcal{C}_3 = \{o_2, o_3, o_9\}$, $\mathcal{C}_4 = \{o_2, o_7\}$, $\mathcal{C}_5 = \{o_4, o_5, o_7\}$, $\mathcal{C}_6 = \{o_6, o_5, o_7\}$ and $\mathcal{C}_7 = \{o_4, o_5, o_6\}$. Because some objects are within more than one coverage sets (e.g., o_2 in $\mathcal{C}_3$ and $\mathcal{C}_4$), many coverage sets are not part of the result set of the *GLS* query. If some coverage sets – which are not necessary for the query result – are refined, the efficiency can be improved considerably. Thus, we present some lemmas that help us in refining some coverage sets as follows.

Algorithm 2: Location Selection Based on Local Coverage Algorithm

Input : $\{< p_j, \mathcal{C}_k > | \mathcal{O} = \bigcup_{k=1}^{T} \mathcal{C}_k\}$
Output: result set $\mathcal{R}$

```
i ⟵ 1
for each object o_i in O do
    for each coverage set o_i ∈ C_k do
        find another coverage set o_i ∈ C_j
    if |C_k| < |C_j| then
        C_k ⟵ C_j
take C_k into R
if ⋃C_k = O then
    take the feasible location for C_k into R
```

Lemma 6: Let K and K' be two subsets of $\{1, 2, \ldots, T\}$. If $\bigcup_{k \in K} \mathcal{C}_k \subseteq \bigcup_{k' \in K'} \mathcal{C}_{k'} \wedge |K| \geq |K'|$, then each $\mathcal{C}_k (k \in K)$ can be refined safely for *GLS* queries.

Proof: For $|K|$ coverage sets generated by the first phase in the framework, any object $o_i \in \bigcup_{k \in K} \mathcal{C}_k$. Therefore, o_i can be influenced by one of $|K|$ temporary facilities that are set up at the feasible locations for these coverage sets. Since $\bigcup_{k \in K} \mathcal{C}_k \subseteq \bigcup_{k' \in K'} \mathcal{C}_{k'} \wedge |K| \geq |K'|$, $o_i \bigcup_{k' \in K'} \mathcal{C}_{k'}$. Note that o_i is also influenced by m' temporary facilities. Due to $|K| \geq |K'|$, the probabilities of these uncertain objects being influenced by a lower number of facilities are not less than τ. Therefore, for *GLS* queries, each $\mathcal{C}_k (k \in K)$ can be refined safely. □

From the above refinement lemma, we can refine coverage sets to improve query efficiency before selecting a minimal number of coverage sets. In Figure 9, $\mathcal{C}_5 \bigcup \mathcal{C}_6 \subseteq \mathcal{C}_4 \bigcup \mathcal{C}_7$, so $\mathcal{C}_5$ and $\mathcal{C}_6$ can be refined safely. Nevertheless, it is still time consuming to cope with the remaining uncertain objects. If any coverage set is definitely chosen by a *GLS* query, we can directly include the coverage set into the result set and consequently reduce the number of coverage sets which further need to be handled. We propose a lemma for this case of efficiency improvement as follows.

Lemma 7: Given a *GLS* query, if an uncertain object o_i is contained only within one coverage set $\mathcal{C}_k$, then

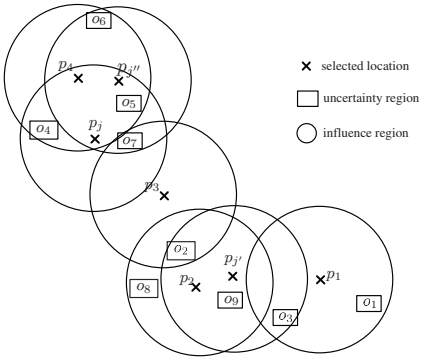


Fig. 9. An example of chosen coverage sets

$\mathcal{C}_k$ is definitely part of the result of the *GLS* query.

Proof: Suppose that a facility located at p can influence an uncertain object o_i with a probability not less than the threshold τ. Since o_i is only within one coverage set $\mathcal{C}_k$, all objects influenced by a facility located at p are certain to be within $\mathcal{C}_k$ with probabilities not less than τ. Thus, $\mathcal{C}_k$ is chosen by the *GLS* query definitely. □

We can use the above lemma to improve efficiency by taking any uncertain objects satisfying this lemma into a coverage set that is chosen by a query directly without complex computations. In our example, the objects o_1 and o_8 are within coverage sets $\mathcal{C}_1$ and $\mathcal{C}_2$, respectively. As illustrated in Figure 9, the two objects can be influenced by facilities located at solely p_1 and p_2. Thus, $\mathcal{C}_1$ and $\mathcal{C}_2$ are refined, and the time cost of *GLS* queries will be reduced.

Next, we present the global coverage algorithm for retrieving the minimum number of coverage sets. For a *GLS* query $GLS(\mathcal{O}, r, \tau)$, the global coverage algorithm decomposes the query into two sub-problems: evaluating the sub-query $GLS(\mathcal{O}', r, \tau)$ ($\mathcal{O}' \subseteq \mathcal{O}$) and selecting $|K|$ coverage sets which contain objects whose number is not less than the number of objects in $|K|$ other coverage sets (i.e., at least a coverage set is different). When $|K|$ coverage sets contain all the objects, $|K|$ is minimal. This process is executed with a Dynamic Programming approach defined as

$$GLS(\mathcal{O}, r, \tau) = GLS(\mathcal{O}', r, \tau) \bigcup (\cup_{k \in K} \{< p_j, \mathcal{C}_k >\}), \quad (8)$$

where $\cup_{k \in K} \{\langle p_j, \mathcal{C}_k \rangle\}$ denotes $|K|$ coverage sets containing the maximum number of objects. From Eq. (8), the algorithm ensures a minimal $|K|$ (w.g., in Figure 9, $\mathcal{O} = \mathcal{C}_1 \bigcup \mathcal{C}_2 \bigcup \mathcal{C}_5 \bigcup \mathcal{C}_6$ or $\mathcal{C}_1 \bigcup \mathcal{C}_2 \bigcup \mathcal{C}_4 \bigcup \mathcal{C}_7$), because we cannot find a lower number (less than four) of *coverage sets* containing all the objects. Thus, the global coverage algorithm can find the minimal number of coverage sets, as detailed in Algorithm 3.

Algorithm 3: Location Selection Based on Global Coverage Algorithm

Input : $\{< p_j, \mathcal{C}_k > | \mathcal{O} = \bigcup_{k=1}^{T} \mathcal{C}_k\}$
Output: result set $\mathcal{R}$

```
1  i ⟵ 1 and 𝒳 ⟵ Φ
2  for each object o_i in 𝒪 do
3      take o_i into a set 𝒳
4      if 𝒳 ⊆ ∪_{k=1}^{j} 𝒞_k then
5          if r > k then
6              r ⟵ k
7          i ⟵ i + 1
8      |K| ⟵ r
9  while k ≤ |K| do
10     take 𝒞_k into ℛ
11     k ⟵ k + 1
```

5.3 Theoretical Analysis

The proposed algorithms of our framework to answer *GLS* queries effectively include a sampling method. Hence, it is important to analyze the probability that the result of any *GLS* query is correct when using the framework, as formalized in the following lemma.

Lemma 8: Let Sr_k be a set of feasible locations for a coverage set $\mathcal{C}_k$ ($k \in K$) chosen by the global coverage algorithm, let S_k be the sample region for $\mathcal{C}_k$, and M be the number of samples. Then, the probability that the result of any *GLS* query is correct when using the framework to answer this query equals

$$\prod_{k \in K} (1 - (1 - \frac{Sr_k}{S_k})^M). \quad (9)$$

Proof: According to Lemma 3, the probability that at least one key sample is a feasible location for a coverage set $\mathcal{C}_k$ is equal to $(1 - (1 - \frac{Sr_k}{S_k})^M)$. Because the global coverage algorithm chooses $|K|$ coverage sets containing all the objects, the probability that a key sample is a feasible location for each chosen coverage set is $\prod_{k \in K}(1 - (1 - \frac{Sr_k}{S_k})^M)$. Since $|K|$ is the minimal number of coverage sets containing all the objects, all the chosen coverage sets are correct for the *GLS* query with a probability that is equal to $\prod_{k \in K}(1 - (1 - \frac{Sr_k}{S_k})^M)$. Therefore, the probability that the result of any *GLS* query is correct when using the framework to answer this query equals $\prod_{k \in K}(1 - (1 - \frac{Sr_k}{S_k})^M)$. □

TABLE 3
Time complexities of algorithms

Algorithm	The time complexity
LCA	$O(n + T)$
GCA	$O(T^2)$
GCA+Refinement	$O(T'^2 + log\ n)$

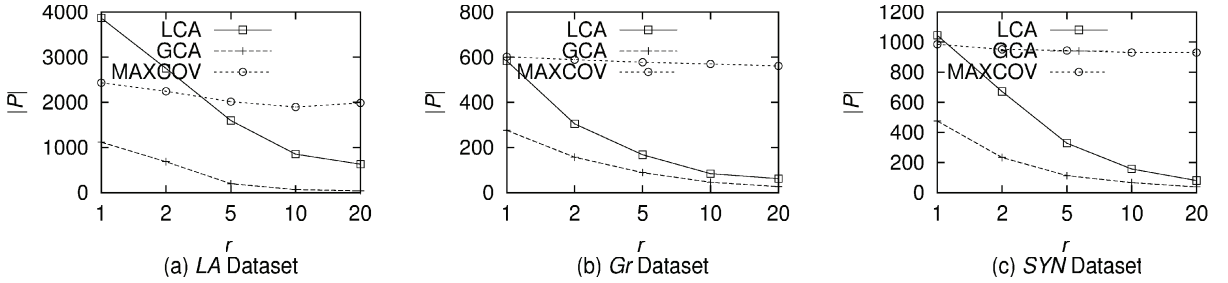


Fig. 10. Effect of r on $|\mathcal{P}|$ with different algorithms

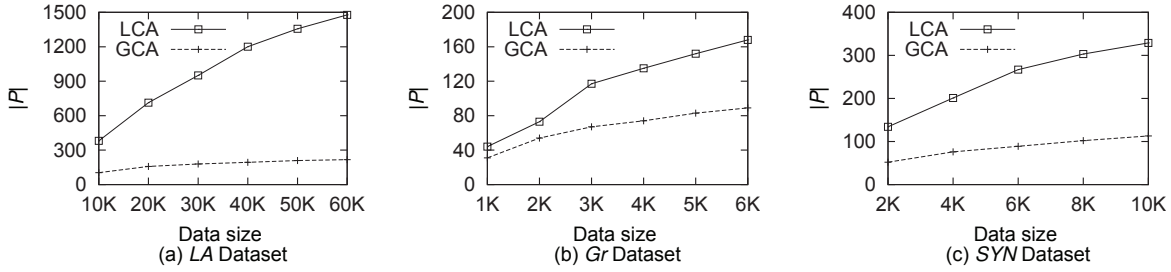


Fig. 11. Effect of data size on $|\mathcal{P}|$ with different algorithms

The above lemma states that our query framework can guarantee that the results of any *GLS* queries are correct with a bounded probability when generating a proper number of samples. Next, we evaluate the time efficiency of the proposed algorithms in the framework. In the first phase of our query framework, we estimate the time cost of the naive solution and the closure polygon algorithm. Suppose that the size of $\mathcal{O}$ is n and the uncertain object set $\mathcal{O}$ is partitioned into T coverage sets in the two-dimensional Euclidean space $\mathcal{U}$. Due to the uncertainty of data, some other objects are contained by the same coverage sets with probabilities that the objects can be influenced by the same facility not less than a given probability threshold τ. Because the naive solution has to evaluate the distance between any two uncertain objects in $\mathcal{O}$, it decides each coverage set and its feasible location one-by-one, which is solved in $O(CMn^2)$ time complexity, where C denotes the average cost of the integral computations. On the other hand, the closure polygon algorithm implements our sampling method for the set of uncertain objects satisfying Lemma 4. The closure polygon algorithm needs to compute integral computations whose number is considerably reduced. Thus, the time complexity of this algorithm is much lower than that of the naive solution. For the selecting phase, the time costs of the local coverage algorithm (LCA), the global coverage algorithm (GCA) and the global coverage algorithm with refinement (GCA+Refinement) are evaluated in Table 3. For all the uncertain objects, the LCA chooses some coverage sets from all the coverage sets so that the time complexity is $O(n + T)$. Compared to this, the Global coverage algorithm answers *GLS* queries with a Dynamic Programming method, which costs $O(T^2)$ time to choose all the coverage sets. The refinements reduce the number of coverage sets to T'. The refinement costs $O(log\ n)$ time and thus the overall cost of the GCA+Refinement is $O(T'^2 + log\ n)$.

6 Experiments

6.1 Experimental Settings

All the experiments were conducted on a PC with a 2.6GHz Intel Processor and 2GB main memory. We used two real world datasets "Los Angeles (*LA*)"and "Greece (*Gr*)", which were also used in [21]. Moreover, one synthetic dataset (*SYN*) was generated with a uniform distribution in the two-dimensional space. We used an R-tree to index all these datasets. The details of these three datasets are as follows.

***LA* Dataset:** The *LA* dataset is a two-dimensional real dataset composed of $60K$ geographical objects described by ranges of longitudes and latitudes.

***Gr* Dataset:** The *Gr* dataset contains 5,922 cities and villages in Greece. For the cities and villages we used a Gaussian distribution for the PDFs to represent them.

***SYN* Dataset:** The *SYN* dataset is composed of $10K$ uncertain objects. Each object was generated by varying ranges of longitudes and latitudes and we used a uniform distribution for the PDFs of objects.

In all the experiments we used RANDLIB [22] to generate M random numbers following the PDFs of uncertain objects and uniform distributions in the intersection regions. In the *LA* dataset, all the objects have the same distribution. For any dataset, the distance between two objects is measured using a Euclidean metric, and we standardized the distances between any two points into the range [0, 1000].

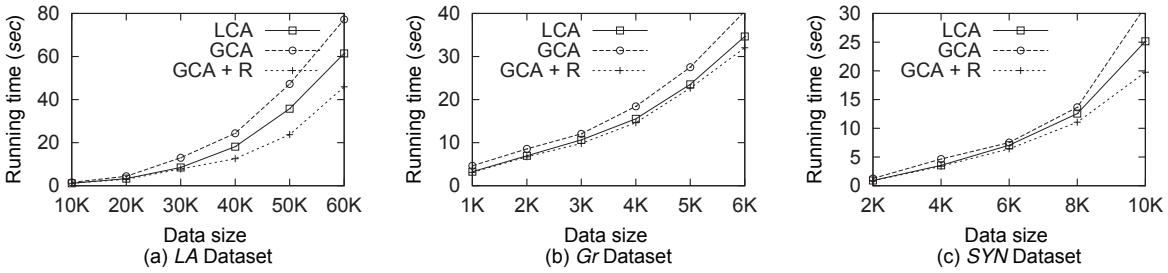


Fig. 12. Effect of data size on the running time with different algorithms

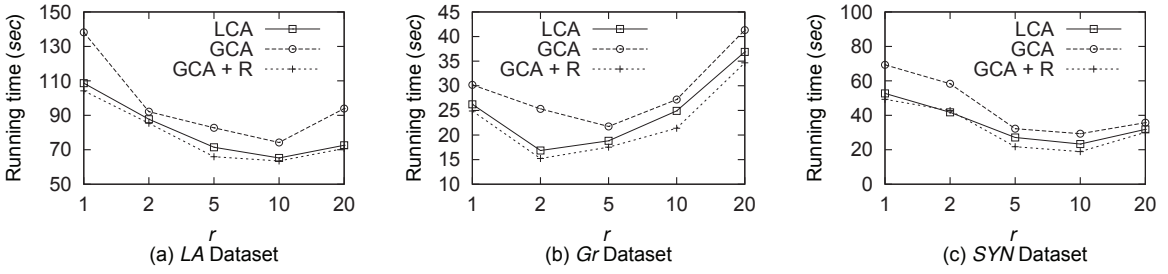


Fig. 13. Effect of r on the running time with different algorithms

TABLE 4
The number of selected locations

	(I) Total number of coverage sets	(II) Number of selected locations with the LCA	(III) Number of selected locations with the GCA
0.1K	673	28	16
0.2K	1383	41	22
0.3K	1625	56	34
0.4K	2014	78	47
0.5K	2458	94	59
0.6K	2974	116	67
0.7K	3367	128	82
0.8K	3920	144	96
0.9K	4278	161	104
1K	4598	172	121

Alternative Techniques Considered. We compare our algorithms with the MAXCOV criterion proposed in [2] with the same three datasets. The MAXCOV criterion can maximize the number of potential objects for setting up facilities, which is seen as a greedy step to set up each facility influencing the maximum number of objects that are closest to this facility (with probabilities that are not less than the threshold).

6.2 Experimental Results

We first analyze the number of selected locations with different methods for a group of uncertain objects in the *SYN* dataset. We use a uniform distribution for the PDFs of these uncertain objects. We fix the radius r of each temporary facility's influence region at 5 and the probability threshold τ at 0.7. Table 4 illustrates the number of total coverage sets generated by the closure polygon algorithm (CPA) and the number of selected locations produced with different methods. Column I shows that the number of total coverage sets increases when the size of the uncertain object set increases. Columns II and III illustrate the number of selected locations selected by the LCA and the GCA. We recognize that the LCA chooses a larger number of selected locations, and that this algorithm only produces a local solution. On the other hand, the GCA chooses a minimal number of selected locations. Hence, the number of selected locations of the GCA is much lower than that of the LCA.

Next we investigate the running time and the number of selected locations while ensuring that the probability of each uncertain object being influenced is not less than τ for all the uncertain objects in the datasets. At first, we compare the number of selected locations denoted by $|\mathcal{P}|$ for the three datasets. As shown in Figure 10, the GCA can influence these objects by building the minimal number of temporary facilities for any dataset. Moreover, for our algorithms, the number of selected locations decreases with increasing r. However, the number of locations with the MAXCOV criterion reduces quite slowly. This is because the MAXCOV criterion considers the objects that are closest to each site, which has a less significant effect on the number of selected locations.

Shown in Figure 11 we compare the number of selected locations denoted by $|\mathcal{P}|$ for different algorithms. Clearly, the GCA can cover these objects with a lower number of selected locations than the LCA for all datasets. The reason is that the GCA chooses a local coverage which overall needs a higher number

of selected locations to influence all the objects with probabilities not less than the threshold τ.

We also study effects of the datasets' size on the time performance of the LCA and the GCA. For the test datasets, we use a Gaussian distribution or a uniform distribution as the PDFs of the uncertain objects, respectively. Figure 12 shows the running time of the LCA, the GCA and the GCA+Refinement as a function of the three dataset sizes. In each case the running time of these algorithms increases with an increasing size of the dataset. The time cost of the LCA is smaller than that of the GCA. However, GCA+Refinement is more efficient than the two algorithms. This demonstrates that the refinement methods improve the efficiency of answering *GIS* queries.

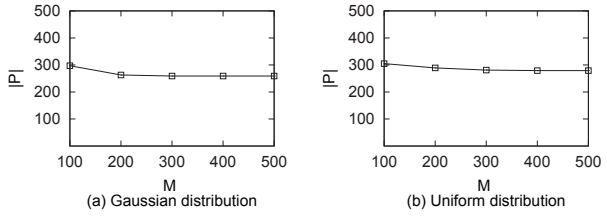


Fig. 14. Effect of M on $|\mathcal{P}|$

We examine the effects of r of each facility's influence region on the running time with the three datasets. As shown in Figure 13, for all the datasets the GCA+Refinement is more efficient than the others. When r rises from 1 to 20, the running time first decreases, but then increases. The efficiency of these algorithms is lower with a proper r, since a larger r results in more uncertain objects for each coverage set and a smaller r generates more coverage sets.

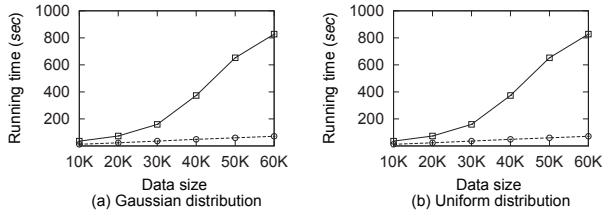


Fig. 15. Comparison with CPA

Subsequently, we test the effects of M on the number of selected locations for Gaussian and uniform object distributions. Figure 14 shows the results where the number of selected locations first decreases and then becomes steady. The reason is that our query framework can find the minimal number of selected locations when M is large enough.

Also, we investigate the running time of the closure polygon algorithm (CPA) and the naive solution (from Section 4) for different probability distributions. The closure polygon algorithm computes the probability that an uncertain object is influenced with Eq. (5) in which M random numbers following a Gaussian or a uniform distribution are generated by RANDLIB [22]. As shown in Figure 15, the running time increases with an increase in the data size. Because the closure polygon algorithm evaluates a lower number of objects and calculates the probabilities that objects are influenced, its running time is much more efficient than that of the naive solution.

We next test the performance of our refinement methods with the three datasets. The refinement steps can refine the massive coverage sets before passing the processing to our other algorithms. As shown in Figure 16, our refinement methods reduce the number of coverage sets for all the datasets effectively. In all cases the reduction rate is 30% or higher.

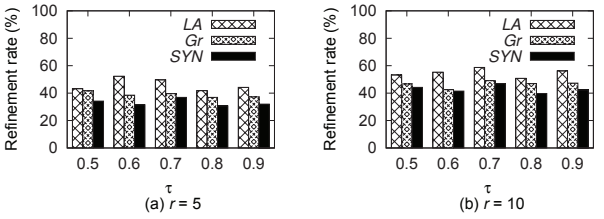


Fig. 16. The refinement rate of coverage sets

Finally, we examine the time cost of answering *GLS* queries over deterministic objects (denoted by points) with the proposed framework. If all the objects are denoted by points, it is possible to improve the efficiency by simplifying our framework. Our strategy is to omit the sampling process. The sampling method in the proposed framework is used to find coverage sets for uncertain objects, which is unnecessary for the case of point objects. Figure 17 illustrates the running time of *GLS* queries with the proposed sampling framework and with the simplified point framework. Clearly, the time cost for our simplified framework is more efficient. The reason is that in the simplified framework we can directly select a proper location to set up a facility for influencing uncertain objects in a coverage set without sampling.

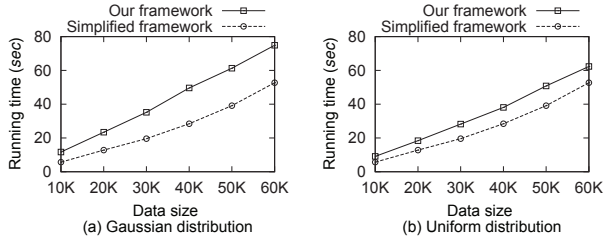


Fig. 17. Analysis for the case of point objects

7 Conclusions

We have proposed a new type of location selection query called *Group Location Selection* (*GLS*) queries. Given a set of uncertain objects, these queries are able to return the minimal number of feasible locations such that setting up a facility at each feasible location will ensure that all objects are influenced. Due to the uncertainty of object locations, we need to ensure that

the probability that each object is influenced by a facility is not less than a given threshold. To answer *GLS* queries efficiently, we presented a general query framework that approximately solves this NP-hard problem with some high-performance methods while guaranteeing that the results of the queries are correct with a bounded probability. In the framework, we first proposed a sampling approach to find feasible locations for all coverage sets efficiently. We then designed the global coverage algorithm to select the minimal number of coverage sets containing all the objects. Finally, we studied the performance of the proposed methods through theoretical analysis and extensive experiments with real and synthetic datasets.

Acknowledgments. The research is supported by the National Basic Research Program of China (973 Program) under Grant No. 2012CB316201, the National Natural Science Foundation of China (61003058, 61033007).

REFERENCES

[1] D. Zhang, Y. Du, T. Xia, and Y. Tao. Progressive computation of the min-dist optimal-location query. In VLDB, pages 643–654, 2006.
[2] S. Cabello, J. M. Díaz-Báñez, S. Langerman, C. Seara, I. Ventura. Reverse facility location problems. In CCCG, pages 68-71, 2005.
[3] X. Xiao, B. Yao, F. Li. Optimal location queries in road network databases. In ICDE, pages 804-815, 2011.
[4] Y. Du, D. Zhang, and T. Xia. The Optimal-Location Query. In Symposium on Large Spatial Databases, pages 163–180, 2005.
[5] R. C.-W. Wong, M. T. Özsu,P. S. Yu, Philip, A. W.-C. Fu, L. Liu. Efficient method for maximizing bichromatic reverse nearest neighbor. In VLDB, pages 1126-1137, 2009.
[6] Z. Zhou, W. Wu, X. Li, M.-L. Lee, W, Hsu. MaxFirst for MaxBRkNN. In ICDE, pages 828-839, 2011.
[7] L. Chen, M. Özsu, V. Oria. Robust and fast similarity search for moving object trajectories. In SIGMOD, pages 491–502, 2005.
[8] R. Cheng, D. V. Kalashnikov, S.Prabhakar. Querying Imprecise Data in Moving Object Environments. In IEEE Trans. on Knowl. and Data Eng., vol. 16, pages 1112–1127, 2004.
[9] Y. Tao, X. Xiao, R. Cheng. Range search on multidimensional uncertain data. In ACM Trans. Database Syst., 2007.
[10] T. H. Cormen, C. E. Leiserson, R. L. Rivest, C. Stein. Introduction to Algorithms, 1989.
[11] S. A. Prasad, W. Ouri, C. Sam, D. Son. Querying the Uncertain Position of Moving Objects. In Temporal Databases: Research and Practice, pages 310–337, 1998.
[12] Y. Ishikawa, Y. Iijima, J.X. Yu. Spatial Range Querying for Gaussian-Based Imprecise Query Objects. In ICDE, pages 676–687, 2009.
[13] T. Xia, D. Zhang, E. Kanoulas, and Y. Du. On computing top-t most influential spatial sites. In VLDB, pages 946–957, 2005.
[14] J. Qi, R. Zhang, L. Kulik, D. Lin, and X. Yuan. The Min-dist Location Selection Query. In ICDE, pages 366–377, 2011.
[15] L. H. U, M. L. Yiu, K. Mouratidis, and N. Mamoulis. Capacity constrained assignment in spatial databases. In SIGMOD, pages 15–28, 2008.
[16] L. H. U, K. Mouratidis, and N. Mamoulis. Continuous spatial assignment of moving users. In VLDB J., vol. 19, pages 141–160, 2010.
[17] A. Caprara, M. Fischetti, and P. Toth. A Heuristic Method for the Set Covering Problem. In Oper. Res., pages 730–743, 1999.
[18] T. A. Feo and M. G. Resende. A probabilistic heuristic for a computationally difficult set covering problem. In Oper. Res., pages 67-71, 1989.
[19] B. Kao, S. D. Lee, F. K. F. Lee, D. W. Cheung, and W.-S. Ho. Clustering Uncertain Data Using Voronoi Diagrams and R-Tree Index. In IEEE Trans. on Knowl. and Data Eng., vol. 22, pages 1219–1233, 2010.
[20] M. R. Garey and D. S. Johnson. Computers and intractability: a guide to theory of Np-completeness. In W.H.Freeman, pages 189-201, 1979.
[21] United States Census Bureau. Topologically Integrated Geographic Encoding and Referencing (TIGER) system. http://www.census.gov/geo/www/tiger/.
[22] RANDLIB. Library of routines for random number generation. https://biostatistics.mdanderson.org/SoftwareDownload/.

Chuanfei Xu received the B.E. degree in 2007 from Shenyang University of Technology, the M.E. degree in 2009 from Computer Science, Northeastern University. Currently, he is a Ph.D candidate in Computer Software and Theory, Northeastern University, China. His research interests include spatial database management and uncertain data management.

Yu Gu received his B.E., M.E., and Ph.D degree in Computer Software and Theory from Northeastern University of China, in 2004, 2007 and 2010, respectively. Currently, he is an associate professor in Computer Software and Theory, Northeastern University, China. His current area of research is spatial database management and graph data management.

Roger Zimmermann received the B.E. degree from the Höhere Technische Lehranstalt in Brugg-Windisch, Aargau, Switzerland, in 1986, and the M.S. and Ph.D degrees in computer science from the University of Southern California, Los Angeles, in 1994 and 1998, respectively. He is currently an associate professor in the Department of Computer Science at the National University of Singapore. His research interests are in the areas of distributed and peer-to-peer systems, streaming media architectures, GIS and location-based services. He is a member of the ACM. He is a senior member of the IEEE.

Shukuan Lin received her B.E. and M.E. degree from Jilin University, Ph.D degree in Computer Software and Theory from Northeastern University of China, in 2006. Currently, she is a professor in the Northeastern University, China. Her research interests include database management, data mining and spatial data management.

Ge Yu received his B.E. degree and M.E. degree in Computer Science from Northeastern University of China in 1982 and 1986, respectively, Ph.D degree in Computer Science from Kyushu University of Japan in 1996. He has been a professor at Northeastern University of China since 1996. His research interests include database theory, distributed and parallel systems, embedded software. He is a senior member of the IEEE.