Data Acquisition for Probabilistic Nearest-Neighbor Query

Yu-Chieh Lin, De-Nian Yang, *Senior Member, IEEE*,
Hong-Han Shuai, and Ming-Syan Chen, *Fellow, IEEE*

Abstract—Management of uncertain data in spatial queries has drawn extensive research interests to consider the granularity of devices and noises in the collection and the delivery of data. Most previous works usually model and handle uncertain data to find the required results directly. However, it is more difficult for users to obtain useful insights when data uncertainty dramatically increases. In this case, users are usually willing to invest more resources to improve the result by reducing the data uncertainty in order to obtain more interesting observations with the existing schemes. In light of this important need, this paper formulates a new problem of selecting a given number of uncertain data objects for acquiring their attribute values to improve the result of the Probabilistic *k*-Nearest-Neighbor (*k*-PNN) query. We prove that better query results are guaranteed to be returned with data acquisition, and we devise several algorithms to maximize the expected improvement. We first explore the optimal single-object acquisition for 1-PNN to examine the fundamental problem structure and then propose an efficient algorithm that discovers crucial properties to simplify the probability derivation in varied situations. We extend the proposed algorithm to achieve the optimal multi-object acquisition for 1-PNN by deriving an upper bound to facilitate efficient pruning of unnecessary sets of objects. Moreover, for data acquisition of *k*-PNN, we extract the *k*-PNN answers with sufficiently large probabilities to trim the search space and properly exploit the result of single-object acquisition for estimating the gain from multi-object acquisition. The experimental results demonstrate that the probability of *k*-PNN can be significantly improved even with only a small number of objects for data acquisition.

Index Terms—Uncertainty, algorithm design and analysis, query processing, nearest neighbor searches

1 Introduction

Management of uncertain data in query processing has attracted extensive research interests in the past few years [2], [13], [14], [28], [31]. In most existing works, authors first formulate a probabilistic problem model for uncertain data and then solve the problem accordingly, by finding the solution with the maximum probability or with the probability no smaller than a threshold. Cheng et al. [13] reduce the errors for range query and nearest-neighbor query with the proposed indexing in moving object environments. Cheng et al. [14] define the probabilistic threshold query and propose an R-tree-based index structure to find the result. Soliman et al. [28] model the probabilistic top-k query as a state space search problem, for jointly considering the traditional top-k semantics and possible worlds semantics. Zhang et al. [31] define the rank based k-NN query considering both expected rank and median rank. Agarwal et al. [2] define the expected nearest neighbor query and build an index of near-linear size to answer it efficiently.

Among all types of queries, the k-nearest-neighbor (k-NN) query is one of the fundamental queries in database management. For uncertain data, the probabilistic k-nearest-neighbor query (k-PNN) is to find the probability for each possible answer set in such a way that the objects in the set are closest to the query point. Since a set of objects with a large joint probability is inclined to bring useful insights, a threshold on the probability is usually incorporated in the problem design to filter out unimportant k-NN answers for the corresponding applications [10].

With only the current uncertain problem models and solution approaches, it is difficult for a user to acquire useful insights and make correct decisions from the answer sets if their k-PNN probabilities are not sufficiently large, especially when the data uncertainty dramatically increases. In this situation, the crux of the problem is the diversity on input uncertain data, instead of the corresponding query processing techniques. Facing this difficulty, users are usually willing to improve the accuracy of a few data objects according to their available resources in hand. For a sensor network, the values of sensors are uncertain because of sampling, in order to reduce the power consumption. Nevertheless, the exact values of specific sensors can be acquired by forcing the sensors to return the current values (i.e., probing) [37]. For a movie rating database, a user may rate the same movie twice with different names (e.g., "Disney's Snow White" and "Snow White"), leading to uncertain rating. Nevertheless, the rating can be revised if the database

- *Y.-C. Lin is with Research Center of Information Technology Innovation, Academia Sinica, Taipei, Taiwan. E-mail: yuccalin@arbor.ee.ntu.edu.tw.*
- *D.-N. Yang is with Institute of Information Science and Research Center of Information Technology Innovation, Academia Sinica, Taipei, Taiwan. E-mail: dnyang@iis.sinica.edu.tw.*
- *H.-H. Shuai is with the Graduate Institute of Communication Engineering, National Taiwan University, Taipei, Taiwan. E-mail: iamshuai@gmail.com.*
- *M.-S. Chen is with Department of Electrical Engineering, National Taiwan University, Taipei, Taiwan, and Research Center of Information Technology Innovation, Academia Sinica, Taipei, Taiwan. E-mail: mschen@cc.ee.ntu.edu.tw.*

Manuscript received 23 Dec. 2012; revised 28 Nov. 2013; accepted 11 Dec. 2013. Date of publication 8 Jan. 2014; date of current version 23 Dec. 2014. Recommended for acceptance by J. Pei.
For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org, and reference the Digital Object Identifier below.
Digital Object Identifier no. 10.1109/TKDE.2013.2297916

TABLE 1
$P(A_{(3,j)} = \{O_x\})$ of Example Data Set in Fig. 1

	$A_{(3,1)}$	$A_{(3,2)}$	$A_{(3,3)}$
$\{O_1\}$	0	0	0.00768
$\{O_2\}$	0	**0.448**	**0.46592**
$\{O_3\}$	**0.8**	0.064	0
$\{O_4\}$	0	0.128	0.1664
$\{O_5\}$	0	0.096	0.096
$\{O_6\}$	0	0.064	0.064
$\{O_7\}$	0.2	0.2	0.2

TABLE 2
$P(A_{(7,j)} = \{O_x\})$ of Example Data Set in Fig. 1

	$A_{(7,1)}$	$A_{(7,2)}$	$A_{(7,3)}$
$\{O_1\}$	0	0.00288	0.00288
$\{O_2\}$	0	**0.39872**	**0.39872**
$\{O_3\}$	0	0.332	0.332
$\{O_4\}$	0	0.1264	0.1264
$\{O_5\}$	0	0.084	0.084
$\{O_6\}$	0	0.056	0.056
$\{O_7\}$	**1**	0	0

system contacts the user for clarification [12]. For a location-based service system, a user's position can be modeled as a set of possible locations when the user is at an indoor place. Nevertheless, the exact user position is possible to be acquired when the system retrieves extra context information from other sensors equipped in mobile handsets such as ambient sound, light, color, and acceleration [34], [35]. Apparently, in most situations it is infeasible to improve the accuracy of all data objects at once due to limited resources available.

Aiming to address the important need above, we propose the notion of data acquisition for the k-PNN query to reduce the uncertainty of a specified number of data objects according to the available resources. As an initial attempt to tackle this realistic but challenging issue, we first explore and focus on a fundamental model with the exact values of those data objects able to be acquired.[1] An intuitive approach is to explore all possible selections and derive the corresponding probabilities. However, this approach is computationally intensive, especially when there are a large number of data objects with each object having considerable probabilities in diverse attribute values. In this paper, therefore, we formulate a new problem, named the *s-acquisition for k-PNN* problem. Given the probability distributions of data objects, the problem is to select a specified number, s, of objects for data acquisition to maximize the improvement of the k-PNN result.

We first prove that every uncertain k-PNN result can be improved by selecting proper data objects for acquisition. To effectively prune the search space, we identify a candidate region of data selection and prove that every object not in the region has no chance to improve the result. Afterward, we explore the optimal 1-acquisition for 1-PNN to examine the fundamental problem structure. We observe that $d_{max}^{(1)}$, which is the smallest maximum distance to the query point among all objects, plays a key role in choosing the objects for data acquisition. Moreover, we systematically derive the probabilities after data acquisition of different objects with the proposed *NAPTable*. After discovering crucial properties to simplify the probability derivation in varied situations, we propose an efficient algorithm to solve the 1-acquisition for 1-PNN. Afterward, we extend the proposed algorithm to achieve the optimal s-acquisition for 1-PNN. We derive an upper bound Q_{MAX} for acquisition of a set of objects and design a *set tree* to find the optimal order of object access and to facilitate efficient pruning of unnecessary sets of objects. Moreover, we design an efficient heuristic algorithm for the data acquisition of k-PNN. To effectively trim the search space, the idea is to systematically extract the *major answers*, which are the k-PNN answers with sufficiently large probabilities, and to exploit the result of single-object acquisition for estimating the effect of multi-object acquisition. The experimental results demonstrate that the probability can be effectively improved with only a limited number of objects for data acquisition.

The remainder of this paper is organized as follows. Related works are introduced in Section 2. In Section 3, we formally define the problem and identify the candidate objects for data acquisition. The problem focusing on the single-object acquisition for 1-PNN is presented in Section 4. In Section 5, we extend the proposed algorithm to solve the multi-object acquisition for 1-PNN. In Section 6, we propose an efficient heuristic algorithm for data acquisition of k-PNN. We present experimental results in Section 7 and conclude the paper in Section 8.

2 Related Work

For uncertain data, a probabilistic database [1] consisting of all possible instances with their probabilities is proposed to model the uncertainty. For a large amount of data, simplified models, such as probabilistic tables [16] and probabilistic or-set tables [24], are developed to solve the scalability issue. Furthermore, the design of query and data mining algorithms for uncertain data has drawn increasing interests. For example, algorithms for probabilistic top-k queries [28], [29], [40], threshold queries [14], aggregate queries [5], k-selection queries [23], and ranking queries [21] are devised to identify the objects with the qualified scores based on pre-defined functions. Algorithms for probabilistic nearest-neighbor queries [6], [9], [20] are proposed to find the objects with the largest probability of being the nearest one to a given query point. Besides, efficient algorithms have been developed to decide the rank-based k nearest neighbors [31], to search the expected nearest neighbor [2], to find the superseding nearest neighbor [30], and to answer the reverse nearest neighbor query [22], the top-k influential query [32], and the nearest neighbor query in probabilistic graphs [26]. Also, related issues on frequent pattern mining [33], skyline queries [7], [19], and clustering [3], [18] with uncertain data have been extensively studied.

For uncertain data collected from the environment with errors and noises, previous works [8], [12], [25] study cleaning processes to identify inadequate data

1. Appendix D, which can be found on the Computer Society Digital Library at http://doi.ieeecomputersociety.org/10.1109/TKDE.2013.2297916, explores the case that data acquisition is not able to accurately identify an exact attribute value. In this case, a small range of values, instead of a single exact value, is obtained after coarse-grained data acquisition.

TABLE 3
Notations Introduced in Section 3

O_i	the i-th object in uncertain database
$v_{i,j}$	the j-th attribute value of O_i
$p_{i,j}$	the probability of $v_{i,j}$
l_i	the number of $v_{i,j}$ of O_i
A	a possible answer of the k-PNN query
$\overline{A}$	the representative answer of the k-PNN query
Q	quality of the k-PNN result
S	a selection set of s objects
$A_{(i_1,j_1)(i_2,j_2)\ldots(i_s,j_s)}$	a new answer of the k-PNN query when $(v_{i_1,j_1},\ldots,v_{i_s,j_s})$ are acquired
$\overline{A}_{(i_1,j_1)(i_2,j_2)\ldots(i_s,j_s)}$	the new representative answer of the k-PNN query when $(v_{i_1,j_1},\ldots,v_{i_s,j_s})$ are acquired
$Q(S)$	average quality of selecting S for data acquisition
$d_{i,j}$	distance between query point and $v_{i,j}$
$d_{i,max}$	the largest $d_{i,j}$ of O_i
$d_{i,min}$	the smallest $d_{i,j}$ of O_i
$d_{\max}^{(k)}$	the k-th $d_{i,max}$ in ascending order
$d_{\min}^{(k)}$	the k-th $d_{i,min}$ in ascending order
C	the set of all candidate objects
R_i	the region from $d_{i,min}$ to $d_{i,max}$

TABLE 4
Procedure: $Single1PNN$

Input: Candidates C
Output: The object for 1-acquisition

Initialize $Index$ as (i,j) pairs where $O_i \in C$ and $d_{i,j} \le d_{max}^{(1)}$
$Index^S \leftarrow Sort(Index)$ {based on $d_{i,j}$ (ascending order)}
Initialize $NAPTable$ of size $n \times n$ with 0's
Initialize $RestP$ of size n with 1's
Initialize Q of size n with 0's
$(PriorX, PriorP, PriorRestP) \leftarrow (1,1,1)$
for all $(i,j) \in Index^S$ **do**
  $X_{i,j} \leftarrow PriorX \times \frac{PriorRestP}{PriorP} \times \frac{p_{i,j}}{RestP[i]}$
  $RestP[i] \leftarrow RestP[i] - p_{i,j}$
  $(PriorX, PriorP, PriorRestP) \leftarrow (X_{i,j}, p_{i,j}, RestP[i])$
  $Q[i] \leftarrow Q[i] + \max(X_{i,j}, (p_{i,j} \times \max_x NAPTable[i][x]))$
  for all $O_a \in C$, $a \neq i$ **do**
    $NAPTable[a][i] \leftarrow NAPTable[a][i] + \frac{X_{i,j}}{RestP[a]}$
  end for
end for{Current Q is stored as Q_B for $Multi1PNN$}
for all O_i **do**
  $Q[i] \leftarrow Q[i] + RestP[i] \times \max_x NAPTable[i][x]$
end for{The 2nd term is stored as $Q_R[i]$ for $Multi1PNN$}
return $\arg\max_{O_i} Q[i]$

objects for any query algorithm. In contrast, this paper considers a different scenario that users are further willing to improve the result by acquiring the correct attribute values of a specified number of data objects according to available resources. In this case, the probability distributions of multiple data objects need to be simultaneously examined to identify the best objects for data acquisition. We specifically focus on the k-PNN query, which is one of the fundamental queries in data management. Instead of random sampling techniques, we propose deterministic algorithms to improve the k-PNN result by maximizing the probabilities of the answer sets.

In [11], authors define a quality metric to evaluate the uncertainty of query results and then provide strategies for data cleaning. The entropy-based *Reynold's metric* in [11] calculates the overall uncertainty of all possible query answers. However, we define the quality of the k-PNN result as the probability of the most probable k-PNN answer, where a larger quality leads to a more definite k-PNN result and can provide more insights. Besides, the goal of data cleaning in [11] is to reduce overall uncertainty on all possible answers by maximizing the expected Reynold's metric after data cleaning. In contrast, the goal of data acquisition is to return a more decisive query answer and does not aim to reduce the uncertainty of other possible answers. Thus, our objective is to maximize the expected quality of the k-PNN result after data acquisition.

3 Problem Description

In this section, we first describe the problem and then demonstrate that acquiring the correct attribute values of a set of data objects can improve the quality of the k-PNN result.

3.1 Problem Definition

Given an uncertain database D with n objects, $\{O_1, O_2, \ldots, O_n\}$, each uncertain object O_i has l_i possible attribute values with the corresponding probabilities,

$$O_i = \{(v_{i,1}, p_{i,1}), (v_{i,2}, p_{i,2}), \ldots, (v_{i,l_i}, p_{i,l_i})\},$$

where $v_{i,j}$ is the jth attribute value of O_i with probability $p_{i,j}$, and $\sum_{j=1}^{l_i} p_{i,j} = 1$.[2] Notice that in this paper, same as the previous works on k-NN query or Top-k query [26], [38], we incorporate a tie-break mechanism in the literature to deal with the tied distances.[3] Tables 3, 5, and 11 list the notations used in the paper.

Definition 1. *Given an uncertain object set D and a query point q, probabilistic k-nearest-neighbor (k-PNN) query [10] is to find the probabilities of all* possible answers. *A possible answer A is defined as an unordered subset of D with cardinality k such that the objects in A are the k nearest neighbors to q with a non-zero probability. That is, in at least one possible world, a possible answer A is the set containing k objects that are the k nearest objects to q, and the order of the k objects in A is not considered.*

In other words, a k-PNN result contains multiple possible answers. According to the definition of k-PNN query [10], the k-PNN query is to evaluate the k-NN query on uncertain databases. The k-NN query is to find the k nearest neighbors to the query point q, but the order of the answering objects is not important. Similarly, the k-PNN query is to find the k nearest neighbors to q in different possible worlds, without considering the order of objects, which is different from uncertain top-k queries such as U-Top$k$ query [28]. Besides, the k-PNN query provides different results comparing to the PT-k query [40], since the goal of the PT-k query is to find every *tuple* with a sufficiently large probability to be included in any top-k answer. More specifically, some tuples returned by a PT-k query may not be simultaneously in the top-k answer in any possible world, due to the individual examination on each tuple.

2. The generalized case that each object has a continuous probability density function is discussed in Appendix D, available in the online supplemental material.

3. If two distances are tied, we differentiate them using a tie-breaker scheme [38]. More specifically, let ϵ denote the distance much smaller than the minimum difference between two distances. If t distance values are tied at d, the scheme assigns new distance values to them as $\{d, d+\frac{\epsilon}{t}, \ldots, d+\frac{(t-1)\epsilon}{t}\}$.

TABLE 5
Notations Introduced in Sections 4 and 5

$X_{i,j}$	probability that $d_{i,j}$ is acquired and O_i is the 1-PNN answer
$d_{i',j'}$	the largest distance that is smaller than $d_{i,j}$
d_{i,j_R}	the smallest distance that is larger than $d^{(1)}_{\max}$ of O_i
P_{Ri}	probability that O_i's distance is larger than $d^{(1)}_{\max}$
$d_{i,\hat{j}}$	the largest distance that is smaller than $d_{i,j}$ among the objects in S
d_{i_R,j_R}	the smallest distance that is larger than $d^{(1)}_{\max}$ among the objects in S
P_{R_S}	probability that the distances of all objects in S are larger than $d^{(1)}_{\max}$
$E_{S(i,j)}$	the set of $(d_{i_1,j_1},\ldots,d_{i_s,j_s})$ combinations with the same smallest acquired distance $d_{i,j}$
$Q_{S(i,j)}$	sum of the propabilities of the representative answers from the same $E_{S(i,j)}$
$A_{S(i,j)}$	the new answer $A_{(i_1,j_1)\ldots(i_s,j_s)}$ when any $(d_{i_1,j_1},\ldots,d_{i_s,j_s})$ combination in $E_{S(i,j)}$ is acquired
$\overline{A}_{S(i,j)}$	the new representative answer $\overline{A}_{(i_1,j_1)\ldots(i_s,j_s)}$ when any $(d_{i_1,j_1},\ldots,d_{i_s,j_s})$ combination in $E_{S(i,j)}$ is acquired
$Q_B(O_i)$	sum of $p_{i,j}\times P(\overline{A}_{(i,j)})$ of each $d_{i,j}\le d^{(1)}_{\max}$ of O_i
$Q_R(O_i)$	sum of $p_{i,j}\times P(\overline{A}_{(i,j)})$ of each $d_{i,j}> d^{(1)}_{\max}$ of O_i
$Q_{MAX}(S)$	upper bound of $Q(S)$

A user would like to receive a k-PNN answer only if its probability is sufficiently large [10]. Besides, when a query result is not acceptable due to high uncertainty, the user in this case usually desires a more definite query result [4]. These prompt us to exploit data acquisition to raise the probability of the received k-PNN answer, especially the largest one. Inspired by the above, we consequently define the quality of the k-PNN result as the probability of the most probable k-PNN answer.

Definition 2. *The quality of the k-PNN result, represented by Q, is defined as*

$$Q=\max_{A}\sum_{1\le j_1\le l_1}\sum_{1\le j_2\le l_2}\cdots\sum_{1\le j_n\le l_n} P(A\wedge(v_{1,j_1},v_{2,j_2},\ldots,v_{n,j_n})),$$

where $(v_{1,j_1},v_{2,j_2},\ldots,v_{n,j_n})$ is defined as an instance *of the uncertain database with probability $\prod_{1\le i\le n}p_{i,j_i}$. If A is the k-NN answer in this instance, $P(A\wedge(v_{1,j_1},v_{2,j_2},\ldots,v_{n,j_n}))=\prod_{1\le i\le n}p_{i,j_i}$. Otherwise, $P(A\wedge(v_{1,j_1},v_{2,j_2},\ldots,v_{n,j_n}))=0$. The answer with the largest probability is defined as the* representative answer $\overline{A}$. *In other words, $P(\overline{A})=Q$.*

The definition is based on the idea that if the most probable answer enjoys a larger probability, the k-PNN result is more definite and can provide more insights. When the quality Q of the k-PNN result is not sufficient to provide useful insights in real applications, it is desirable to invest additional resources and select a set S of s objects $\{O_{i_1},O_{i_2},\ldots,O_{i_s}\}$ to acquire their attribute values such that the quality can be further improved. In this paper, we assume that the exact value of an uncertain object can be acquired according to the literature.[4] In other words, it is assumed that the object after data acquisition is with no uncertainty.

Definition 3. *Given a set S containing s objects, if their attribute values are acquired as $(v_{i_1,j_1},v_{i_2,j_2},\ldots,v_{i_s,j_s})$, let $A_{(i_1,j_1)(i_2,j_2)\ldots(i_s,j_s)}$ denote a new answer to the k-PNN query, and let $\overline{A}_{(i_1,j_1)(i_2,j_2)\ldots(i_s,j_s)}$ denote the new representative answer with the largest probability.*

The new $\overline{A}_{(i_1,j_1)(i_2,j_2)\ldots(i_s,j_s)}$ can be different from the original $\overline{A}$ because the attribute values of the objects in S are no longer uncertain after data acquisition. However, the challenge of data acquisition lies in the fact that the attribute values $(v_{i_1,j_1},v_{i_2,j_2},\ldots,v_{i_s,j_s})$ are unknown during the selection of the s objects. Thus, it is necessary to examine the representative answer $\overline{A}_{(i_1,j_1)(i_2,j_2)\ldots(i_s,j_s)}$ of different attribute values $(v_{i_1,j_1},v_{i_2,j_2},\ldots,v_{i_s,j_s})$ with the corresponding probabilities $(p_{i_1,j_1},p_{i_2,j_2},\ldots,p_{i_s,j_s})$ to find the *average quality* of S.

Definition 4. *The average quality $Q(S)$ of selecting a set S for data acquisition is*

$$Q(S)=\sum_{1\le j_1\le l_{i_1}}\sum_{1\le j_2\le l_{i_2}}\cdots\sum_{1\le j_s\le l_{i_s}} P(\overline{A}_{(i_1,j_1)(i_2,j_2)\ldots(i_s,j_s)})p_{i_1,j_1}p_{i_2,j_2}\ldots p_{i_s,j_s},$$

where $\overline{A}_{(i_1,j_1)(i_2,j_2)\ldots(i_s,j_s)}$ can vary for different attribute values of the objects in S.

According to Definition 2, $Q=P(\overline{A})$ is the original quality of the k-PNN result, and in Definition 4, $Q(S)$ is the average quality of the k-PNN result after acquiring the attribute values of objects in S. Lemma 1 demonstrates that the average quality of selecting any object O_i never decreases.

Lemma 1. *For any object O_i, $Q(\{O_i\})\ge Q$.*

Proof. For any acquired $v_{i,j}$ of O_i, if $\overline{A}_{(i,j)}\neq\overline{A}$, we have $P(\overline{A}_{(i,j)})>P(\overline{A}|v_{i,j})$. Otherwise, i.e., $\overline{A}_{(i,j)}=\overline{A}$, we have $P(\overline{A}_{(i,j)})=P(\overline{A}|v_{i,j})$. In summary, $Q(\{O_i\})=\sum_{\forall j}(P(\overline{A}_{(i,j)})\times p_{i,j})\ge\sum_{\forall j}(P(\overline{A}|v_{i,j})\times p_{i,j})=P(\overline{A})=Q$. The lemma follows. □

Therefore, it is desirable to select an S that generates the largest average quality of the k-PNN result to effectively invest additional resources. With this objective in mind, we formulate the following problem for this paper.

Definition 5. *The problem s-*acquisition for k-PNN *is to select a set S of s objects such that $Q(S)$ is maximized.*

3.2 Candidate Identification

Lemma 2. *For each object O_i, $Q(\{O_i\})>Q$ if $\overline{A}_{(i,r)}\neq\overline{A}$ holds for any attribute value $v_{i,r}$.*[5]

To improve the average quality, it is important to avoid the case with the new representative answer $\overline{A}_{(i,j)}$ identical to the original $\overline{A}$. To guide an efficient search of S that leads to the largest $Q(S)$, it is desirable to first remove the data objects that are never able to generate a different new representative answer for all possible attribute values.

4. The exact values of specific sensors can be probed in a sensor network [37]. Besides, the exact position of a user can be extracted from a set of possible locations by retrieving extra information from sensors equipped in mobile handsets [34], [35].

5. Due to space constraint, the proofs of some lemmas and theorems in the rest of this paper are presented in Appendix E, available in the online supplemental material.

With this objective in mind, for each object O_i, let $d_{i,j}$ denote the distance between query point q and attribute value $v_{i,j}$, where $d_{i,\max} = \max_{1 \le j \le l_i} d_{i,j}$ and $d_{i,\min} = \min_{1 \le j \le l_i} d_{i,j}$. We sort $d_{i,\max}$ in ascending order and let $d_{\max}^{(k)}$ denote the kth distance. Meanwhile, we sort $d_{i,\min}$ in ascending order and let $d_{\min}^{(k)}$ denote the kth distance. Lemmas 3 and 4 identify two categories of objects that are never able to generate a new representative answer for all possible attribute values. The first category is the objects with all attribute values very close to q, while the second category includes the objects with all attribute values far away from q. Furthermore, Lemma 5 demonstrates a case that no further quality improvement can be achieved by data acquisition.

Lemma 3. *For each object O_i, if $d_{i,\max} < d_{\min}^{(k+1)}$, then $\overline{A}_{(i,j)} = \overline{A}$, $1 \le j \le l_i$.*

Lemma 4. *For each object O_i, if $d_{i,\min} > d_{\max}^{(k)}$, then $\overline{A}_{(i,j)} = \overline{A}$, $1 \le j \le l_i$.*

Lemma 5. *If $d_{\max}^{(k)} \le d_{\min}^{(k+1)}$, no object can increase the quality of the k-PNN result.*

With the above observations, Theorem 1 identifies the objects required to be examined for data acquisition. Specifically, let R_i of object O_i denote the region from $d_{i,\min}$ to $d_{i,\max}$.

Theorem 1. *If the quality can be improved by data acquisition, there exists at least one object O_i such that R_i overlaps with the region from $d_{\min}^{(k+1)}$ to $d_{\max}^{(k)}$.*

Proof. Lemma 5 shows that the quality of the k-PNN result can never be improved if $d_{\max}^{(k)} \le d_{\min}^{(k+1)}$. Therefore, we target on the condition of $d_{\min}^{(k+1)} < d_{\max}^{(k)}$. In this case, if the attribute values of every object are either all smaller than $d_{\min}^{(k+1)}$ or all larger than $d_{\max}^{(k)}$, Lemmas 3 and 4 show that the quality still cannot be improved, respectively. The theorem follows. □

Theorem 1 demonstrates that the region between $d_{\min}^{(k+1)}$ and $d_{\max}^{(k)}$ is a key factor to improve the quality. The objects located in other regions are not necessary to be considered.

Definition 6. *The* candidate region *of the k-PNN query is from $d_{\min}^{(k+1)}$ to $d_{\max}^{(k)}$, and every object O_i with R_i overlapping with the candidate region is a* candidate object *for data acquisition.*

It is worth noting that we exploit the candidate region for candidate identification, which is different from the pruning phase in [13] using only $d_{\max}^{(1)}$ as the filter or the k-bound filtering in [10] using only $d_{\max}^{(k)}$ as the filter. Based on Definition 6, we filter the candidate objects as follows.

The candidate set C is initialized with all objects in it. Rather than sorting, we first scan the database once to find the $d_{\min}^{(k+1)}$ and $d_{\max}^{(k)}$ by updating two ordering lists. During the second scan, from C we remove every object that is not in the candidate region. When the algorithm terminates, the objects remaining in C are the candidate objects.

More specifically, an ordering list of size k is used to record the smallest k of all scanned $d_{i,\max}$ values in ascending order. We denote a snapshot of the ordering list as $(d_{\max}^{\prime(1)}, d_{\max}^{\prime(2)}, \ldots, d_{\max}^{\prime(k)})$. When an object O_i is scanned, $d_{i,\max}$ is compared with the current $d_{\max}^{\prime(k)}$. If $d_{i,\max} < d_{\max}^{\prime(k)}$, $d_{i,\max}$ is inserted into the ordering list. Similarly, another ordering list of size $k+1$ is used to record the smallest $k+1$ of all scanned $d_{i,\min}$ values in ascending order. After the first database scan, $d_{\max}^{(k)}$ and $d_{\min}^{(k+1)}$ are identified. According to Lemma 5, if $d_{\max}^{(k)} < d_{\min}^{(k+1)}$, there is no acquisition for k-PNN. Hence, we remove all objects from C and terminate the whole procedure. Else, for each object in C, we decide whether it can be removed according to Lemma 3 or Lemma 4. Since the objects removed according to Lemma 3 must belong to every k-PNN answer, the k value is decreased by the number of these objects. That is, the k value considered in following algorithms could be smaller than the original k. Without loss of generality, we assume that the k value is not decreased after the candidate identification process in this paper.

4 Single-Object Acquisition for 1-PNN

After truncating unnecessary objects that are outside the candidate region, the next is to find the optimal selection set of objects for data acquisition to maximize the quality. A straightforward approach is to sequentially examine the average quality $Q(S)$ for every possible selection set S of objects. However, if each object has l possible values, there are l^s possible sets of values for the s objects in each possible S. It is required to identify the representative answer for each set of values for the s objects. Moreover, the computation becomes more intensive as k increases. Therefore, to efficiently process this challenging problem, Section 4 first solves the fundamental problem of data acquisition with $s = 1$ and $k = 1$ to explore important properties. Based on the properties observed, general cases will then be considered in Sections 5 and 6.

4.1 Problem Analysis

For a 1-PNN query, the result contains each answer $\{O_i\}$ with $P(A = \{O_i\}) > 0$. To select an object for data acquisition, a simple approach is to extract the one with the largest probability to further improve the quality. However, this approach, though simple, is not able to reach the maximum quality because the quality is also correlated to the data distribution of other objects, as illustrated in the following example. Each object O_i of the data set in Fig. 1 has three possible distances, denoted as $d_{i,1}, d_{i,2}, d_{i,3}$ in ascending order, and the corresponding probabilities $p_{i,1}, p_{i,2}, p_{i,3}$ are labeled beside the nodes. In this example, O_2 has the largest probability $P(A = \{O_2\}) = 0.319$ with quality $Q(\{O_2\}) = 0.435$. However, selecting O_2 for data acquisition fails to generate the maximum quality because O_3 is a better choice with a larger quality $Q(\{O_3\}) = 0.559$ while having a smaller probability $P(A = \{O_3\}) = 0.266$. The example illustrates that a more sophisticated approach is desirable even for this fundamental problem of data acquisition. In the following, we first explore several crucial properties in the derivation of quality to significantly reduce the computational overhead. Equipped with those properties, an efficient algorithm is then proposed to optimally select the object for data acquisition.

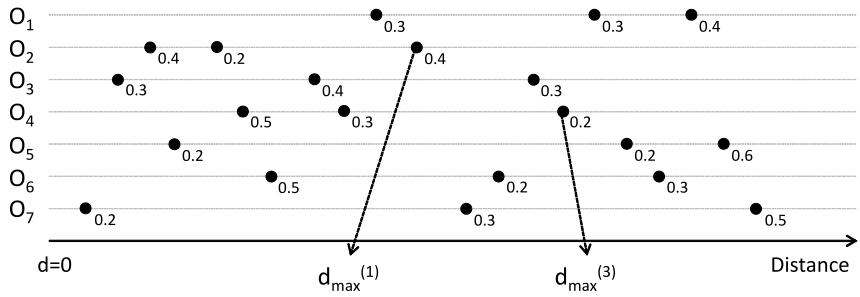

Fig. 1. An example for algorithm design of data acquisition for 1-PNN and k-PNN.

Following Definition 4, the average quality of selecting any single object $\{O_i\}$ is

$$Q(\{O_i\}) = \sum_{j=1}^{l_i} (p_{i,j} \times P(\overline{A}_{(i,j)})) = \sum_{j=1}^{l_i} \max_x (p_{i,j} \times P(A_{(i,j)} = \{O_x\})). \quad (1)$$

To obtain the best 1-acquisition for 1-PNN, $Q(\{O_i\})$ of each $\{O_i\}$ is derived to identify the one with the largest average quality. The quality $Q(\{O_i\})$ is the sum of $p_{i,j} \times P(\overline{A}_{(i,j)})$ for each representative answer, which is decided after $P(A_{(i,j)} = \{O_x\})$ for each x is calculated. During the examination of instances in $P(A_{(i,j)} = \{O_x\})$, it is not sufficient to find out objects with only $d_x < d_{i,j}$, where d_x represents the distance from object O_x to the query point q. We also need to ensure that d_x is smaller than all the other objects.

To achieve the above goal, when $d_{i,j}$ is acquired, it is intuitive to obtain $P(A_{(i,j)})$ by summing up the probabilities for all combinations of $(j_1, \ldots, j_{i-1}, j_{i+1}, \ldots, j_n)$. For example, $P(A_{(3,2)} = \{O_3\})$ is obtained after summing up the probabilities $P(\{d_{1,1} \vee d_{1,2} \vee d_{1,3}\}, d_{2,3}, \{d_{4,2} \vee d_{4,3}\}, \{d_{5,2} \vee d_{5,3}\}, \{d_{6,2} \vee d_{6,3}\}, \{d_{7,2} \vee d_{7,3}\})$ of all the 48 combinations that contain no distance smaller than $d_{3,2}$. However, instead of directly finding out the probabilities of the 48 combinations separately, a more efficient approach is to first sum up $P(d_1 > d_{3,2}) = p_{1,1} + p_{1,2} + p_{1,3}$, $P(d_4 > d_{3,2}) = p_{4,2} + p_{4,3}$, ..., and then derive $P(A_{(3,2)} = \{O_3\})$ as the product of $P(d_1 > d_{3,2})$, $P(d_2 > d_{3,2})$, $P(d_4 > d_{3,2})$, , which results in $P(A_{(3,2)} = \{O_3\}) = 1.0 \times 0.4 \times 0.5 \times 0.8 \times 0.5 \times 0.8 = 0.064$. Apparently, this aggregation approach is able to significantly reduce the number of probabilities involved in the above derivation. Therefore, when $d_{i,j}$ is acquired, the new 1-PNN answer could be either $\{O_i\}$ itself with probability

$$P(A_{(i,j)} = \{O_i\}) = \prod_{\forall h | O_h \in C, h \neq i} P(d_h > d_{i,j}),$$

or any other object $\{O_x\}$ with probability

$$P(A_{(i,j)} = \{O_x\}) = \sum_{\forall y | d_{x,y} < d_{i,j}} \Big(p_{x,y} \times \prod_{\forall h | O_h \in C, h \neq x, h \neq i} P(d_h > d_{x,y}) \Big). \quad (2)$$

To further improve the efficiency, we simplify the derivation of each probability $P(A_{(i,j)})$ by properly exploiting other probabilities derived before. Specifically, for any $d_{x,y} < d_{i,j}$ acquired for calculating $Q(\{O_x\})$, the new answer could be $\{O_x\}$ itself with probability $P(A_{(x,y)} = \{O_x\}) = \prod_{\forall h | O_h \in C, h \neq x} P(d_h > d_{x,y})$. It is worth noting that $\prod_{\forall h | O_h \in C, h \neq x} P(d_h > d_{x,y})$ of $P(A_{(x,y)} = \{O_x\})$ is very similar to $\prod_{\forall h | O_h \in C, h \neq x, h \neq i} P(d_h > d_{x,y})$ of $P(A_{(i,j)} = \{O_x\})$. Therefore, after $\prod_{\forall h | O_h \in C, h \neq x} P(d_h > d_{x,y})$ of some $d_{x,y}$ with $d_{x,y} < d_{i,j}$ is derived, the probability can be employed to find out the $\prod_{\forall h | O_h \in C, h \neq x, h \neq i} P(d_h > d_{x,y})$ of $P(A_{(i,j)} = \{O_x\})$ as $P(A_{(x,y)} = \{O_x\})/P(d_i > d_{x,y})$. To formally facilitate the above observation, we define $X_{i,j}$ as follows.

Definition 7. *$X_{i,j}$ is defined as*

$$X_{i,j} = p_{i,j} \times P(A_{(i,j)} = \{O_i\}) = p_{i,j} \times \prod_{\forall h | O_h \in C, h \neq i} P(d_h > d_{i,j}),$$

representing the joint probability that $d_{i,j}$ is acquired and $\{O_i\}$ is the 1-PNN answer.

We exploit $X_{i,j}$ to efficiently process two cases with $d_{i,j} \leq d_{max}^{(1)}$ and $d_{i,j} > d_{max}^{(1)}$, respectively. Before studying other interesting properties, Fig. 2 presents the flow chart of the proposed algorithm for single-object acquisition. Recall that in Eq. (1), $Q(\{O_i\})$ is the sum of $p_{i,j} \times P(\overline{A}_{(i,j)})$ of each j. In the following, we will first demonstrate that $p_{i,j} \times P(\overline{A}_{(i,j)})$ is the maximum of $X_{i,j}$ and $p_{i,j} \times P(A_{(i,j)} = \{O_x\})$ for each $x \neq i$ in Lemma 6. We will then exploit the probabilities obtained before to efficiently derive $X_{i,j}$ in Lemma 7 and find out $p_{i,j} \times P(A_{(i,j)} = \{O_x\})$ for each $x \neq i$ in Lemma 8. Finally, for $d_{i,j} > d_{max}^{(1)}$, we demonstrate that $p_{i,j} \times P(\overline{A}_{(i,j)})$ can be considered as a whole to further improve the efficiency according to Lemma 10. After the lemmas are concluded by Theorem 2, the algorithm is then proposed.

4.2 Representative New Answer of $d_{i,j} \leq d_{max}^{(1)}$

Lemma 6. *With Definition 7, the $p_{i,j} \times P(A_{(i,j)} = \{O_x\})$ in Eq. (1) can be rewritten as $X_{i,j}$ for $x = i$, or as*

$$p_{i,j} \times \sum_{\forall y | d_{x,y} < d_{i,j}} \frac{X_{x,y}}{P(d_i > d_{x,y})}$$

for $x \neq i$.

Appendix E, available in the online supplemental material, details the proof of the lemma. Lemma 6 indicates

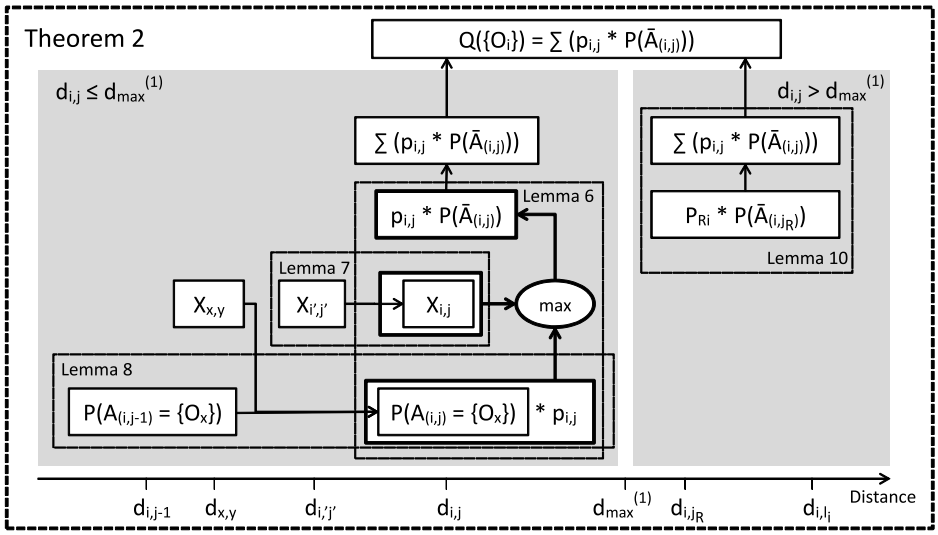

Fig. 2. Flow chart to derive $Q(\{O_i\})$ for each O_i.

that all $X_{x,y}$ of $d_{x,y} < d_{i,j}$ can be efficiently reused in the derivation of $P(A_{(i,j)} = \{O_x\})$. Moreover, $p_{i,j} \times P(\overline{A}_{(i,j)})$ in Fig. 2 is thus the maximum one among $X_{i,j}$ and $p_{i,j} \times P(A_{(i,j)} = \{O_x\})$ for each $x \neq i$. Assuming that $d_{i',j'} < d_{i,j}$ and there exists no object with a distance between $d_{i',j'}$ and $d_{i,j}$, Lemma 7 further proposes to efficiently calculate $X_{i,j}$ with $X_{i',j'}$ obtained before.

Lemma 7. *With $d_{i,j}$ sorted, the relationship between any neighboring $d_{i',j'} < d_{i,j}$ is that*

$$X_{i,j} = X_{i',j'} \times \frac{P(d_{i'} > d_{i,j})}{p_{i',j'}} \times \frac{p_{i,j}}{P(d_i > d_{i',j'})}.$$

According to the relationship, we calculate $X_{i,j}$ in ascending order of the corresponding $d_{i,j}$.

Recall that in Fig. 2, $p_{i,j} \times P(\overline{A}_{(i,j)})$ is the maximum among $X_{i,j}$ and $p_{i,j} \times P(A_{(i,j)} = \{O_x\})$ for each $x \neq i$. After efficiently processing $X_{i,j}$ with $X_{i',j'}$, Lemma 8 simplifies $p_{i,j} \times P(A_{(i,j)} = \{O_x\})$ for each $x \neq i$ by properly exploiting $P(A_{(i,j-1)} = \{O_x\})$ to find out $P(A_{(i,j)} = \{O_x\})$.

Lemma 8.

$$P(A_{(i,j)} = \{O_x\}) = P(A_{(i,j-1)} = \{O_x\}) + \sum_{\forall y | d_{i,j-1} \leq d_{x,y} < d_{i,j}} \frac{X_{x,y}}{P(d_i > d_{x,y})}.$$

Equipped with Lemma 8, we derive $P(A_{(i,j)} = \{O_x\})$ in ascending order of $d_{i,j}$ to efficiently process the cases $d_{i,j} \leq d^{(1)}_{\max}$ and $d_{i,j} > d^{(1)}_{\max}$ in single-object acquisition.

4.3 Representative New Answer of $d_{i,j} > d^{(1)}_{max}$

In the following, we demonstrate that $p_{i,j} \times P(\overline{A}_{(i,j)})$ of the same object O_i can be derived as a whole for all $d_{i,j} > d^{(1)}_{max}$ based on an important observation that $P(A_{(i,j)} = \{O_x\})$ values with the same i and x but different j are identical for $d_{i,j} > d^{(1)}_{\max}$.

Lemma 9. *Assume that $d_{i,j_R-1} < d^{(1)}_{\max} < d_{i,j_R}$, we have $P(A_{(i,j_R)} = \{O_x\}) = P(A_{(i,j_R+1)} = \{O_x\}) = \cdots = P(A_{(i,l_i)} = \{O_x\})$.*

With Lemma 9, for every j with $d_{i,j} > d^{(1)}_{max}$, $P(A_{(i,j)} = \{O_x\})$ is identical to $P(A_{(i,j_R)} = \{O_x\})$. In other words, $P(\overline{A}_{(i,j)} = \{O_x\}) = P(\overline{A}_{(i,j_R)} = \{O_x\})$, which is the same for every j with $d_{i,j} > d^{(1)}_{max}$. This important property enables us to derive the average quality as a whole.

Definition 8. *We define $P_{Ri} = \sum_{\forall j | d_{i,j} > d^{(1)}_{\max}} p_{i,j}$, representing the probability that O_i's distance is larger than $d^{(1)}_{\max}$.*

Lemma 10.

$$\sum_{\forall j | j_R \leq j \leq l_i} (p_{i,j} \times P(\overline{A}_{(i,j)})) = P_{Ri} \times P(\overline{A}_{(i,j_R)}).$$

Equipped with all crucial observations above, we efficiently derive $Q(\{O_i\})$ in Theorem 2.

Theorem 2. *The average quality $Q(\{O_i\})$ can be represented as follows:*

$$Q(\{O_i\}) = \sum_{\forall j | 1 \leq j < j_R} (p_{i,j} \times P(\overline{A}_{(i,j)})) + P_{Ri} \times P(\overline{A}_{(i,j_R)}),$$

where $p_{i,j} \times P(\overline{A}_{(i,j)})$ is the largest among $X_{i,j}$ and $p_{i,j} \times \sum_{\forall y | d_{x,y} < d_{i,j}} \frac{X_{x,y}}{P(d_i > d_{x,y})}$ for all $x \neq i$, and the two terms can be derived from $X_{i',j'}$ and $P(A_{(i,j-1)} = \{O_x\})$, respectively.

Proof. Following Eq. (1),

$$Q(\{O_i\}) = \sum_{\forall j | 1 \leq j < j_R} (p_{i,j} \times P(\overline{A}_{(i,j)})) + \sum_{\forall j | j_R \leq j \leq l_i} (p_{i,j} \times P(\overline{A}_{(i,j)})).$$

With the second term reformulated according to Lemma 10,

$$Q(\{O_i\}) = \sum_{\forall j | 1 \leq j < j_R} (p_{i,j} \times P(\overline{A}_{(i,j)})) + P_{Ri} \times P(\overline{A}_{(i,j_R)}).$$

By definition, $p_{i,j} \times P(\overline{A}_{(i,j)})$ is the largest of every $p_{i,j} \times P(A_{(i,j)}) = \{O_x\}$, which is $X_{i,j}$ for $x = i$ or

$p_{i,j} \times \sum_{\forall y | d_{x,y} < d_{i,j}} \frac{X_{x,y}}{P(d_i > d_{x,y})}$ for $x \neq i$ based on Lemma 6, and the two terms can be derived based on Lemmas 7 and 8, respectively. The theorem follows. □

Example 1. Fig. 1 presents an illustrative example to explain the basic ideas of Lemmas 6-10 and Theorem 2 in Sections 4.2 and 4.3. In single-object acquisition for 1-PNN, the quality of acquiring O_i's exact value is the expected probability of the most probable 1-PNN answer after the exact value of O_i is acquired. Thus, a simple approach is to sequentially examine each O_i for finding the optimal solution

$$Q(\{O_i\}) = \sum_{1 \le j \le 3} \Big(p_{i,j} \times \max_{1 \le x \le 7} P(A_{(i,j)} = \{O_x\}) \Big).$$

Now we take O_3 as an example. The quality of acquiring O_3's exact value is

$$Q(\{O_3\}) = \sum_{1 \le j \le 3} \Big(p_{3,j} \times \max_{1 \le x \le 7} P(A_{(3,j)} = \{O_x\}) \Big).$$

For each j, we need to find $P(A_{(3,j)} = \{O_x\})$ of each $\{O_x\}$ to decide the maximum one. Table 1 lists $P(A_{(3,j)} = \{O_x\})$ for each j and $\{O_x\}$. Therefore, $Q(\{O_3\}) = 0.3 \times 0.8 + 0.4 \times 0.448 + 0.3 \times 0.46592 = 0.558976$.

More specifically, the event $A_{(3,2)} = \{O_2\}$ consists of two cases: (1) $d_{2,1}$ is the smallest distance among all objects, and (2) $d_{2,2}$ is the smallest distance among all objects. (The case that $d_{2,3}$ is the smallest distance cannot happen since $d_{2,3} > d_{3,2}$.) Thus, $P(A_{(3,2)} = \{O_2\})$ is

$$\begin{aligned} P(A_{(3,2)} = \{O_2\}) = p_{2,1} \times \prod_{\forall h | h \notin \{2,3\}} P(d_h > d_{2,1}) \\ + p_{2,2} \times \prod_{\forall h | h \notin \{2,3\}} P(d_h > d_{2,2}), \end{aligned} \quad (3)$$

where the two terms correspond to the two cases, and d_h represents object O_h's distance. We have $P(A_{(3,2)} = \{O_2\}) = 0.4 \times (1.0)(1.0)(1.0)(1.0)(0.8) + 0.2 \times (1.0)(1.0)(0.8)(1.0)(0.8) = 0.448$.

To improve the efficiency, Lemma 6 organizes the above derivation and extracts the probabilities calculated before to avoid duplicated calculations. According to Definition 7, $X_{2,1} = p_{2,1} \times \prod_{\forall h | h \neq 2} P(d_h > d_{2,1})$ and $X_{2,2} = p_{2,2} \times \prod_{\forall h | h \neq 2} P(d_h > d_{2,2})$. Thus, $P(A_{(3,2)} = \{O_2\})$ in Eq. (3) can be rewritten as

$$P(A_{(3,2)} = \{O_2\}) = \frac{X_{2,1}}{P(d_3 > d_{2,1})} + \frac{X_{2,2}}{P(d_3 > d_{2,2})}. \quad (4)$$

Since $X_{2,1}$ has been derived in $\max_{\forall x} P(A_{(2,1)} = \{O_x\})$, here we can directly employ $X_{2,1}$ in $P(A_{(3,2)} = \{O_2\})$. With $X_{2,1} = 0.224$ and $X_{2,2} = 0.0896$, we have $P(A_{(3,2)} = \{O_2\}) = \frac{0.224}{0.7} + \frac{0.0896}{0.7} = 0.448$, as shown in Table 1.

In contrast to directly applying Definition 7, Lemma 7 suggests a more efficient approach to find $X_{i,j}$. Consider $X_{3,2}$ as an example, $X_{3,2} = p_{3,2} \times \prod_{\forall h | h \neq 3} P(d_h > d_{3,2})$. Since $d_{6,1}$ is the largest distance among all objects' possible distances which are smaller than $d_{3,2}$, $P(d_h > d_{3,2}) = P(d_h > d_{6,1})$ holds for $h \neq 3$, and

$$X_{3,2} = p_{3,2} \times P(d_6 > d_{3,2}) \times \prod_{\forall h | h \notin \{3,6\}} P(d_h > d_{6,1}).$$

Multiplied by $\frac{P(d_3 > d_{6,1})}{P(d_3 > d_{6,1})}$ and $\frac{p_{6,1}}{p_{6,1}}$, the equation can be rewritten as

$$X_{3,2} = X_{6,1} \times \frac{P(d_6 > d_{3,2})}{p_{6,1}} \times \frac{p_{3,2}}{P(d_3 > d_{6,1})}.$$

By exploiting $X_{6,1}$ derived in $\max_{\forall x} P(A_{(6,1)} = \{O_x\})$ before, Lemma 7 enables us to find $X_{3,2}$ more efficiently. With $X_{6,1} = 0.0448$, we have $X_{3,2} = X_{6,1} \times \frac{0.5}{0.5} \times \frac{0.4}{0.7} = 0.0256$, instead of $X_{3,2} = 0.4 \times (1.0)(0.4)(0.5)(0.8)(0.5)(0.8)$. It is worth noting that when there are more uncertain objects, more multiplications can be reduced by Lemma 7.

Lemma 6 reuses $X_{x,y}$ to simplify the calculating of $P(A_{(i,j)} = \{O_x\})$. Further, Lemma 8 reuses $P(A_{(i,j-1)} = \{O_x\})$ to find $P(A_{(i,j)} = \{O_x\})$ more efficiently. Consider $P(A_{(3,3)} = \{O_2\})$ as an example. Based on Lemma 6,

$$\begin{aligned} &P(A_{(3,3)} = \{O_2\}) \\ &= \frac{X_{2,1}}{P(d_3 > d_{2,1})} + \frac{X_{2,2}}{P(d_3 > d_{2,2})} + \frac{X_{2,3}}{P(d_3 > d_{2,3})}. \end{aligned}$$

By replacing the first two terms with $P(A_{(3,2)} = \{O_2\})$ in Eq. (4), $P(A_{(3,3)} = \{O_2\})$ is derived according to Lemma 8 as

$$P(A_{(3,3)} = \{O_2\}) = P(A_{(3,2)} = \{O_2\}) + \frac{X_{2,3}}{P(d_3 > d_{2,3})}.$$

By exploiting $P(A_{(3,2)} = \{O_2\}) = 0.448$ found before, we have $P(A_{(3,3)} = \{O_2\}) = 0.448 + 0.01792 = 0.46592$, as shown in Table 1.

If O_i has more than one possible distances exceeding $d_{\max}^{(1)}$, Lemma 9 states that $P(A_{(i,j)} = \{O_x\})$ is the same for every $d_{i,j} > d_{\max}^{(1)}$ of O_i. Take $P(A_{(7,2)} = \{O_4\})$ and $P(A_{(7,3)} = \{O_4\})$ for example. Based on Lemma 8,

$$P(A_{(7,3)} = \{O_4\}) = P(A_{(7,2)} = \{O_4\}) + \frac{X_{4,3}}{P(d_7 > d_{4,3})}.$$

Because $d_{4,3}$ is larger than $d_{2,3}$ (i.e., $d_{\max}^{(1)}$), we have $X_{4,3} = 0$ by definition. Thus, $P(A_{(7,3)} = \{O_4\})$ is the same as $P(A_{(7,2)} = \{O_4\})$. Therefore, if $P(A_{(7,2)} = \{O_4\})$ has been derived before, $P(A_{(7,3)} = \{O_4\})$ can be directly looked up without any calculation.

Referring to Table 2, the quality of acquiring O_7's exact value is $Q(\{O_7\}) = 0.2 \times 1 + 0.3 \times 0.39872 + 0.5 \times 0.39872 = 0.518976$. Since $P(A_{(7,3)} = \{O_x\}) = P(A_{(7,2)} = \{O_x\})$ holds for any O_x, $\max_{\forall x} P(A_{(7,3)} = \{O_x\}) = \max_{\forall x} P(A_{(7,2)} = \{O_x\})$. Thus, Lemma 10 regards $d_{7,2}$ and $d_{7,3}$ as a whole during the calculation, presenting $Q(\{O_7\})$ as $Q(\{O_7\}) = 0.2 \times 1 + (0.3 + 0.8) \times 0.39872 = 0.518976$. It is worth noting that when there are many possible values larger than $d_{\max}^{(1)}$, more multiplications can be avoided by Lemma 10. Finally, Theorem 2 summarizes Lemmas 6-10 and it acts as the formula for single-object acquisition quality.

TABLE 6
The $P(A_{(i,j)} = \{O_x\})$ After $d_{2,1}$ is Examined

	$\{O_1\}$	$\{O_2\}$	$\{O_3\}$	$\{O_4\}$	$\{O_5\}$	$\{O_6\}$	$\{O_7\}$
O_1	—	0.224	0.24	0	0	0	0.2
O_2	0	—	0.24	0	0	0	0.2
O_3	0	0.32	—	0	0	0	0.2
O_4	0	0.224	0.24	—	0	0	0.2
O_5	0	0.224	0.24	0	—	0	0.2
O_6	0	0.224	0.24	0	0	—	0.2
O_7	0	0.224	0.3	0	0	0	—

TABLE 7
The $P(A_{(i,j)} = \{O_x\})$ After $d_{2,3}$ is Examined

	$\{O_1\}$	$\{O_2\}$	$\{O_3\}$	$\{O_4\}$	$\{O_5\}$	$\{O_6\}$	$\{O_7\}$
O_1	—	0.3213	0.2656	0.1011	0.0672	0.0448	0.2
O_2	0.0058	—	0.304	0.2528	0.112	0.112	0.2
O_3	0.0077	0.4659	—	0.1664	0.096	0.064	0.2
O_4	0.0115	0.3405	0.2912	—	0.0672	0.0896	0.2
O_5	0.0029	0.3427	0.272	0.1264	—	0.056	0.2
O_6	0.0046	0.3244	0.2912	0.1126	0.0672	—	0.2
O_7	0.0029	0.3987	0.332	0.1264	0.084	0.056	—

4.4 Algorithm Design

Based on Theorem 2, we propose the *Single1PNN* algorithm to solve the problem of 1-acquisition for 1-PNN as follows. The algorithm contains two phases corresponding to the previous two sections to systematically and efficiently derive $Q(\{O_i\})$ for each i. In the first phase, each acquired distance $d_{i,j}$ not larger than $d^{(1)}_{\max}$ is examined in ascending order according to Lemmas 6, 7, and 8. In the second phase, the quality of each object is updated based on the acquired distance larger than $d^{(1)}_{\max}$ according to Lemma 10. Table 4 presents the pseudocode of the algorithm. The details are explained as follows.

The first phase accumulates $Q(\{O_i\})$ from each distance $d_{i,j}$ with $d_{i,j} \leq d^{(1)}_{\max}$ according to the observations in Section 4.2. The $d_{i,j}$ with $d_{i,j} \leq d^{(1)}_{\max}$ is first sorted in ascending order. For each $d_{i,j}$, we find $P(A_{(i,j)} = \{O_x\})$ for all O_x to derive $p_{i,j} \times P(\overline{A}_{(i,j)})$ following Lemma 6. To store intermediate results, a simple approach is to create a table with three dimensions to record $P(A_{(i,j)} = \{O_x\})$ for each (i, j, x) trio. Nevertheless, by carefully examining $P(A_{(i,j)} = \{O_x\})$ in ascending order of $d_{i,j}$, now only a two-dimensional table is required because for the same O_i and O_x, different j values can sequentially share an identical entry. Specifically, according to Lemma 8, for the same O_i and the same O_x, $P(A_{(i,j+1)} = \{O_x\})$ is derived from $P(A_{(i,j)} = \{O_x\})$. Therefore, for each $d_{i,j}$, after $P(A_{(i,j)} = \{O_x\})$ of all O_x are calculated and $p_{i,j} \times P(\overline{A}_{(i,j)})$ is obtained, we can employ $P(A_{(i,j)} = \{O_x\})$ to calculate $P(A_{(i,j+1)} = \{O_x\})$ following Lemma 8, and $P(A_{(i,j)} = \{O_x\})$ is no longer necessary to be maintained afterward.

With the above observation, *new answer probability table (NAPTable)*, which is a table of size $n \times n$, is created for accumulating the $P(A_{(i,j)} = \{O_x\})$ values for every (i, x). When examining $d_{i,j}$, we obtain $P(A_{(i,j)} = \{O_x\})$ by looking up $NAP\,Table[i][x]$ and immediately update $p_{i,j} \times P(\overline{A}_{(i,j)})$ if necessary. Then, we add $\frac{X_{x,y}}{P(d_i > d_{x,y})}$ to $NAP\,Table[i][x]$ when examining $d_{x,y}$ with $d_{i,j} \leq d_{x,y} < d_{i,j+1}$. Afterward, for $d_{i,j+1}$, $NAP\,Table[i][x]$ now stores $P(A_{(i,j+1)} = \{O_x\})$, and we can look up $NAPTable[i][x]$ to obtain $p_{i,j+1} \times P(\overline{A}_{(i,j+1)})$ accordingly.

With $Q(\{O_i\})$ of each i initialized as 0, we examine each $d_{i,j}$ in ascending order as follows.

1. Calculate $X_{i,j}$ using $X_{i',j'}$ following Lemma 7.
2. Identify $p_{i,j} \times P(\overline{A}_{(i,j)})$ as the largest among $X_{i,j}$ and $p_{i,j} \times P(A_{(i,j)} = \{O_x\})$ for each $x \neq i$ based on Lemma 6 by looking up $P(A_{(i,j)} = \{O_x\})$ from $NAP\,Table[i][x]$. It is worth noting that the value of $NAP\,Table[i][x]$ has been turned from $P(A_{(i,j-1)} = \{O_x\})$ to $P(A_{(i,j)} = \{O_x\})$ after each $d_{x,y}$ with $d_{i,j-1} \leq d_{x,y} < d_{i,j}$ is examined.
3. Add $p_{i,j} \times P(\overline{A}_{(i,j)})$ to the accumulated $Q(\{O_i\})$.
4. Add $\frac{X_{i,j}}{P(d_a > d_{i,j})}$ to $NAP\,Table[a][i]$ for each O_a with $a \neq i$, which is to accumulate the $P(A_{a,b} = \{O_i\})$, based on Lemma 8. To avoid confusion, here O_a represents any object with $O_a \neq O_i$. We look up $P(A_{i,j} = \{O_x\})$ with $x \neq i$ when $d_{i,j}$ is examined, and we will extract $P(A_{a,b} = \{O_i\})$ and $P(A_{a,b} = \{O_x\})$ with $x \neq a$ and $x \neq i$ to decide $p_{a,b} \times P(\overline{A}_{(a,b)})$ when $d_{a,b}$ is examined in the future.

Afterward, the second phase accumulates $Q(\{O_i\})$ from each distance $d_{i,j}$ with $d_{i,j} > d^{(1)}_{\max}$ based on observations in Section 4.3. By looking up $NAPTable[i][x]$, i.e., $P(\overline{A}_{(i,j_R)} = \{O_x\})$, we add $P_{Ri} \times P(\overline{A}_{(i,j_R)} = \{O_x\})$ to $Q(\{O_i\})$ of each O_i based on Lemma 10. With all $Q(\{O_i\})$ updated, the O_i corresponding to the largest $Q(\{O_i\})$ is the optimal solution for 1-acquisition.

Example 2. Consider the data set in Fig. 1. First, all $Q(\{O_i\})$ and $P(A_{(i,j)} = \{O_x\})$ are initialized as zero. In the first phase, $d_{i,j}$ is examined in ascending order starting from $d_{7,1}$.

Table 6 presents $P(A_{(i,j)} = \{O_x\})$ after $d_{7,1}$, $d_{3,1}$, and $d_{2,1}$ are examined, where each row represents an O_i, and each column represents an $\{O_x\}$. When $d_{7,1}$ is examined, $X_{7,1} = 0.2$. $P(A_{(i,j)} = \{O_7\})$ for each $i \neq 7$ is increased by $\frac{X_{7,1}}{1.0}$, and $Q(\{O_7\})$ is added by 0.2. Next, $d_{3,1}$ is examined and $X_{3,1} = 0.24$. $P(A_{(7,j)} = \{O_3\})$ is increased by $\frac{X_{3,1}}{0.8}$, and each of other $P(A_{(i,j)} = \{O_3\})$ is increased by $\frac{X_{3,1}}{1.0}$. With $X_{3,1} > p_{3,1} \times P(A_{(3,1)} = \{O_7\})$, $Q(\{O_3\})$ is added by 0.24. Then, $d_{2,1}$ is examined and $X_{2,1} = 0.224$. $P(A_{(7,j)} = \{O_2\})$ is increased by $\frac{X_{2,1}}{0.8}$ and $P(A_{(3,j)} = \{O_2\})$ is increased by $\frac{X_{2,1}}{0.7}$, while each of other $P(A_{(i,j)} = \{O_2\})$ is increased by $\frac{X_{2,1}}{1.0}$. With $P(A_{(2,1)} = \{O_3\}) > P(A_{(2,1)} = \{O_7\})$, we compare $p_{2,1} \times P(A_{(2,1)} = \{O_3\})$ with $X_{2,1}$. Since $X_{2,1}$ is the larger, $Q(\{O_2\})$ is added by 0.224.

Table 7 demonstrates $P(A_{(i,j)} = \{O_x\})$ and $Q(\{O_i\})$ after $d_{2,3}$ is examined. After $d_{2,3}$, which is $d^{(1)}_{\max}$, is examined, the second phase starts. The first row of Table 8 shows the quality $Q(\{O_i\})$ that has been

TABLE 8
$Q(\{O_i\})$ and P_{Ri} After $d_{2,3}$ is Considered

	O_1	O_2	O_3	O_4	O_5	O_6	O_7
$Q(\{O_i\})$	0.00941	0.4352	0.4192	0.2509	0.0672	0.1568	0.2
P_{Ri}	0.7	0	0.3	0.2	0.8	0.5	0.8

TABLE 9
Final $Q(\{O_i\})$, $Q_B(O_i)$, and $Q_R(O_i)$

	O_1	O_2	O_3	O_4	O_5	O_6	O_7
Q	0.3190	0.4352	0.5590	0.3190	0.3414	0.3190	0.5190
Q_B	0.0941	0.4352	0.4192	0.2509	0.0672	0.1568	0.2
Q_R	0.2249	0	0.1398	0.0681	0.2742	0.1622	0.3190

accumulated from the beginning. With the total of probability of unexamined distances, i.e., P_{Ri}, shown in Table 8, the quality increased by the unexamined distances of each object is then calculated. For example, $Q(\{O_7\})$ is increased by 0.3190, which is $P_{R7} \times P(A_{(7,j)} = \{O_2\})$. With the final average quality of all objects shown in Table 9, the best 1-acquisition for 1-PNN is $\{O_3\}$ with $Q(\{O_3\}) = 0.559$.

5 Multi-Object Acquisition for 1-PNN

Compared with single-object acquisition, the result of 1-PNN can be improved when more resources are invested for s-acquisition. Nevertheless, new challenges in finding the optimal s objects also arise due to more combinations of objects necessary to be carefully examined. Fortunately, $X_{i,j}$ defined in Definition 7 can be utilized in another way for aggregating the average quality of a selected set of objects. Moreover, we exploit the average quality $Q(\{O_i\})$ of selecting a single object O_i to estimate the average quality $Q(S)$ of selecting a set S of s objects, and we develop an efficient pruning strategy to avoid examining every possible set of objects.

5.1 s-Acquisition for 1-PNN

The average quality $Q(S)$ of selecting a set $S = \{O_{i_1}, O_{i_2}, \ldots, O_{i_s}\}$ is

$$Q(S) = \sum_{1 \le j_1 \le l_{i_1}} \sum_{1 \le j_2 \le l_{i_2}} \cdots \sum_{1 \le j_s \le l_{i_s}} P(\overline{A}_{(i_1,j_1)(i_2,j_2)\ldots(i_s,j_s)}) \times p_{i_1,j_1} p_{i_2,j_2} \ldots p_{i_s,j_s}.$$

Compared with 1-acquisition, s-acquisition is more challenging due to the following reasons.

1. Much more possible selections need to be considered. For 1-acquisition, there are only n selections, i.e., $\{O_1\}, \ldots, \{O_n\}$. In contrast, C_s^n selections are involved for s-acquisition.
2. For each selection, there are much more possible acquired values. For a selection $\{O_i\}$ in 1-acquisition, $P(A_{(i,j)}) \times p_{i,j}$ needs to be derived for each distance $d_{i,j}$. For a selected set S in s-acquisition, however, $P(A_{(i_1,j_1)(i_2,j_2)\ldots(i_s,j_s)}) \times p_{i_1,j_1} p_{i_2,j_2} \ldots p_{i_s,j_s}$ is required to be derived for every possible combination $(d_{i_1,j_1}, d_{i_2,j_2}, \ldots, d_{i_s,j_s})$ of acquired distances.

It is worth noting that solving the s-acquisition for 1-PNN belongs to a type of minimum-element problems [36], which are NP-hard.[6]

6. The hardness of the problem is discussed in detail in Appendix F, available in the online supplemental material.

To efficiently find out $Q(S)$, we first show that the probabilities $P(\overline{A}_{(i_1,j_1)(i_2,j_2)\ldots(i_s,j_s)}) \times p_{i_1,j_1} p_{i_2,j_2} \ldots p_{i_s,j_s}$ of multiple $(d_{i_1,j_1}, d_{i_2,j_2}, \ldots, d_{i_s,j_s})$ combinations are the same such that those combinations can be processed as a whole. For each combination $(d_{i_1,j_1}, d_{i_2,j_2}, \ldots, d_{i_s,j_s})$, if the smallest acquired distance in S is $d_{i,j}$, the new answer could be either $\{O_i\}$, with probability

$$P(A_{(i_1,j_1)(i_2,j_2)\ldots(i_s,j_s)} = \{O_i\}) = \prod_{\forall h | O_h \in C - S} P(d_h > d_{i,j}), \quad (5)$$

or any other $\{O_x\}$ not in S, i.e., $O_x \in C - S$, with probability

$$P(A_{(i_1,j_1)(i_2,j_2)\ldots(i_s,j_s)} = \{O_x\}) = \sum_{\forall y | d_{x,y} < d_{i,j}} \left(p_{x,y} \times \prod_{\forall h | O_h \in C - S - \{O_x\}} P(d_h > d_{x,y}) \right). \quad (6)$$

According to the two equations above, if multiple possible combinations of acquired distances $(d_{i_1,j_1}, d_{i_2,j_2}, \ldots, d_{i_s,j_s})$ share the same smallest distance, $P(A_{(i_1,j_1)(i_2,j_2)\ldots(i_s,j_s)})$ of them are the same. Thus, we sum up those probabilities as follows.

Definition 9. *For each S, we define $E_{S(i,j)}$ as the set of $(d_{i_1,j_1}, d_{i_2,j_2}, \ldots, d_{i_s,j_s})$ combinations that have the same smallest acquired distance, $d_{i,j}$. The probability of $E_{S(i,j)}$ is*

$$P(E_{S(i,j)}) = \sum_{\forall (d_{i_1,j_1},\ldots,d_{i_s,j_s}) \in E_{S(i,j)}} p_{i_1,j_1} \times \cdots \times p_{i_s,j_s} = p_{i,j} \times \prod_{\forall h | O_h \in S - \{O_i\}} P(d_h > d_{i,j}).$$

Similarly, we sum up the probabilities of representative answers from the same $E_{S(i,j)}$.

Definition 10. *We define*

$$Q_{S(i,j)} = \sum_{\forall (d_{i_1,j_1},\ldots,d_{i_s,j_s}) \in E_{S(i,j)}} (P(\overline{A}_{(i_1,j_1)(i_2,j_2)\ldots(i_s,j_s)}) \times p_{i_1,j_1} p_{i_2,j_2} \ldots p_{i_s,j_s}),$$

which is the amount contributed *to $Q(S)$ by all of the $(d_{i_1,j_1}, d_{i_2,j_2}, \ldots, d_{i_s,j_s})$ combinations in $E_{S(i,j)}$. Abbreviatedly, we say that $E_{S(i,j)}$* contribute *$Q_{S(i,j)}$ to $Q(S)$.*

Definition 11. *With $E_{S(i,j)}$ in Definition 9, we define $A_{S(i,j)}$ as the new answer $A_{(i_1,j_1)(i_2,j_2)\ldots(i_s,j_s)}$ when any $(d_{i_1,j_1}, d_{i_2,j_2}, \ldots, d_{i_s,j_s})$ combination in $E_{S(i,j)}$ is acquired. Eqs. (5) and (6) can be rewritten as*

$$P(A_{S(i,j)} = \{O_i\}) = \prod_{\forall h | O_h \in C - S} P(d_h > d_{i,j}), \text{ and}$$

$$P(A_{S(i,j)} = \{O_x\}) = \sum_{\forall y | d_{x,y} < d_{i,j}} \left(p_{x,y} \times \prod_{\forall h | O_h \in C - S - \{O_x\}} P(d_h > d_{x,y}) \right).$$

We define $\overline{A}_{S(i,j)}$ as the new representative answer, i.e., the $A_{S(i,j)}$ with the largest probability.

Definition 12. *Let $d_{i_R,\hat{j}_R}$ and d_{i_R,j_R} denote two possibly acquired distances of objects in S with $d_{i_R,\hat{j}_R} < d^{(1)}_{\max} < d_{i_R,j_R}$, and there is no other possibly acquired distance from $d_{i_R,\hat{j}_R}$ to d_{i_R,j_R}. We define*

$$P_{R_S} = \prod_{\forall h|O_h \in S} P(d_h \geq d_{i_R,j_R}) = \sum_{\forall (i,j)|O_i \in S, d_{i,j} \geq d_{i_R,j_R}} P(E_{S(i,j)}), \quad (7)$$

representing the probability that the distances of all objects in S are larger than $d^{(1)}_{\max}$.

Following the same logic flow of the derivation of $Q(\{O_i\})$ in 1-acquisition, we derive Theorem 3 to efficiently calculate $Q(S)$ of a possible selection set S. In Section 4, we sum up $p_{i,j} \times P(\overline{A}_{(i,j)})$ to derive $Q(\{O_i\})$. Similarly, here we sum up $P(E_{S(i,j)}) \times P(\overline{A}_{S(i,j)})$ to derive $Q(S)$. More details of Theorem 3 can be found in Appendix B, available in the online supplemental material.

Theorem 3.

$$Q(S) = \sum_{\forall (i,j)|O_i \in S, d_{i,j} \leq d^{(1)}_{\max}} Q_{S(i,j)} + P_{R_S} \times \max_x P(A_{S(i_R,j_R)} = \{O_x\}),$$

where $Q_{S(i,j)}$ is the largest among $X_{i,j}$ and

$$P(E_{S(i,j)}) \times \sum_{\forall y|d_{x,y} < d_{i,j}} \frac{X_{x,y}}{\prod_{\forall h|O_h \in S} P(d_h > d_{x,y})}$$

of each $O_x \in C - S$. Besides, $P(E_{S(i,j)})$ and $P(A_{S(i,j)} = \{O_x\})$ can be derived from $P(E_{S(i,j)})$ and $P(A_{S(i,j)} = \{O_x\})$, respectively, where $d_{i,j}$ denotes the previous possible acquired distance of objects in S when sorted in ascending order.

5.2 Set Quality Estimation

Although it is efficient to derive $Q(S)$ based on Theorem 3, it is computationally intensive if every possible selection set S is required to be examined to find $Q(S)$ accordingly. In the following, therefore, we first derive an upper bound $Q_{MAX}(S)$ on $Q(S)$ as an estimation, and later the upper bound is exploited for effective pruning of possible selection sets such that only a limited number of S are involved to find the optimal selection for s-acquisition.

We first define Q_B and Q_R, which are related to the average quality of selecting a single object. Then, we employ Q_B and Q_R to derive an upper bound $Q_{MAX}(S)$ on $Q(S)$.

Definition 13. *For each object O_i, we define $Q(\{O_i\}) = Q_B(O_i) + Q_R(O_i)$, where*

$$Q_B(O_i) = \sum_{\forall j|d_{i,j} \leq d^{(1)}_{max}} (p_{i,j} \times P(\overline{A}_{(i,j)})),$$

$$Q_R(O_i) = \sum_{\forall j|d_{i,j} > d^{(1)}_{max}} \big(p_{i,j} \times P(\overline{A}_{(i,j)})\big).$$

We partition $Q(\{O_i\})$ into two parts by $d^{(1)}_{\max}$: $Q_B(O_i)$ and $Q_R(O_i)$. More specifically, $Q_B(O_i)$ is a part of $Q(\{O_i\})$, which is contributed by $d_{i,j} \leq d^{(1)}_{\max}$ of O_i, and $Q_R(O_i)$ is the other part of $Q(\{O_i\})$, which is contributed by $d_{i,j} > d^{(1)}_{\max}$ of O_i.

Definition 14. *For each S, we define*

$$Q_{MAX}(S) = Q_B(O_{i_1}) + Q_B(O_{i_2}) + \cdots + Q_B(O_{i_s}) + \min(Q_R(O_{i_1}), Q_R(O_{i_2}), \ldots, Q_R(O_{i_s})),$$

representing the maximum quality led by acquiring the exact values of the objects in S, as shown in Theorem 4. Every object $O_i \in S$ contributes $Q_B(O_i)$ to $Q_{MAX}(S)$. However, only the object with $\min_{\forall O_i \in S} Q_R(O_i)$ contributes its $Q_R(O_i)$ to $Q_{MAX}(S)$.

Lemma 11. *or each $d_{i,j}$, the average quality contributed by $E_{S(i,j)}$ is no more than $p_{i,j} \times P(\overline{A}_{(i,j)})$, i.e., $Q_{S(i,j)} \leq p_{i,j} \times P(\overline{A}_{(i,j)})$.*

Lemma 12.

$$P_{R_S} \times \max_x \sum_{\forall y|d_{x,y} < d_{i_R,j_R}} \frac{X_{x,y}}{\prod_{\forall h|O_h \in S} P(d_h > d_{x,y})} \leq Q_R(O_i). \quad (8)$$

Theorem 4. *$Q_{MAX}(S)$ is an upper bound of $Q(S)$, i.e., $Q(S) \leq Q_{MAX}(S)$.*

Proof. According to the derivation of $Q(S)$ in Theorem 3,

$$Q(S) = \sum_{\forall (i,j)|O_i \in S, d_{i,j} \leq d^{(1)}_{max}} Q_{S(i,j)} + P_{R_S} \times \max_x \sum_{\forall y|d_{x,y} < d_{i_R,j_R}} \frac{X_{x,y}}{\prod_{\forall h|O_h \in S} P(d_h > d_{x,y})}. \quad (9)$$

Based on Lemma 11 and Definition 13,

$$\sum_{\forall j_1|d_{i_1,j_1} \leq d^{(1)}_{max}} Q_{S(i_1,j_1)} \leq \sum_{\forall j_1|d_{i_1,j_1} \leq d^{(1)}_{max}} (p_{i_1,j_1} \times P(\overline{A}_{(i_1,j_1)})) = Q_B(O_{i_1}).$$

Similar inequalities hold for $O_{i_2}, \ldots, O_{i_s}$. Thus,

$$\sum_{\forall (i,j)|O_i \in S, d_{i,j} \leq d^{(1)}_{\max}} Q_{S(i,j)} \leq Q_B(O_{i_1}) + Q_B(O_{i_2}) + \cdots + Q_B(O_{i_s}). \quad (10)$$

For the second term in Eq. (9), since Eq. (8) holds for any $O_i \in S$, the minimum one of $(Q_R(O_{i_1}), Q_R(O_{i_2}), \ldots, Q_R(O_{i_s}))$ is selected to generate the tightest bound. By summing up Eqs. (10) and (8), we have $Q(S) \leq Q_{MAX}(S)$. The theorem follows. □

Following Theorem 4, for any two sets S and S^* with $Q_{MAX}(S) < Q(S^*)$, $Q(S) < Q(S^*)$ must hold. Therefore, when $Q(S^*)$ represents the best average quality obtained so far, we can first find $Q_{MAX}(S)$ and avoid unnecessary derivation of $Q(S)$ if $Q_{MAX}(S) < Q(S^*)$ holds since S can never substitute S^*. In other words, S^* can be exploited for efficient pruning of any other selection set S, and it is thus desirable to obtain a sufficiently large $Q(S^*)$ at an early stage.

TABLE 10
Procedure: $Multi1PNN$

Input: s, Candidates C, Q_B, Q_R
Output: S^* {The selection set for s-acquisition}
1 Initialize S^* and SPQ empty
2 $Q(S^*) \leftarrow 0$
3 Create roots of set trees and insert them into SPQ
4 Pop out S from SPQ
5 **while** $Q_{MAX}(S) \geq Q(S^*)$ **do**
6 $Q(S) \leftarrow Quality(S)$
7 **if** $Q(S) > Q(S^*)$ **then**
8 $S^* \leftarrow S$
9 $Q(S^*) \leftarrow Q(S)$
10 **end if**
11 Generate child nodes of S and insert them into SPQ
12 Pop out S from SPQ
13 **end while**
14 **return** S^*

5.3 Algorithm Design

Equipped with upper bound $Q_{MAX}(S)$, we devise the *Multi1PNN* algorithm to solve s-acquisition for 1-PNN, which avoids deriving $Q(S)$ for every possible selection set S by exploiting the notion of *set tree* introduced later. To find a good S that can act as S^*, we sort S in descending order of $Q_{MAX}(S)$, because any set S with a high $Q_{MAX}(S)$ is more possible to enjoy a large $Q(S)$ as well. To sort S in descending order of $Q_{MAX}(S)$, a straightforward way is to enumerate all possible selection sets and calculate their $Q_{MAX}(S)$. However, a more efficient way is to exploit a hierarchical data structure and partition all possible selection sets into several groups. The partition is performed to guarantee that $\min Q_R$ of every selection set belonging to the same group corresponds to the same object. Apparently, for each group with the same $\min Q_R$, the selection set S with the largest $Q_{MAX}(S)$ consists of other $s-1$ objects with the largest Q_B by Definition 14. Therefore, it is possible to generate another selection set S' with a smaller $Q_{MAX}(S')$ by replacing an object in S by another object with a smaller Q_B.

Thus, for each group of selection sets with the same $\min Q_R$ corresponding to an object O_i, we define the *set tree* of O_i in order to first extract the selection set S with the largest $Q_{MAX}(S)$ and then systematically explore other selection sets in the same group. Specifically, with carefully designed object replacement rules (e.g., replacing an object at a time, following the descending order of Q_B for replacement, recording the position of the replaced object in the set), all non-repetitive sets containing objects with the same $\min Q_R$ can be generated once on the set tree in accordance with Q_{MAX}.[7] Further, a *Set Priority Queue (SPQ)* is used to store the sets generated from any set tree in descending order of $Q_{MAX}(S)$, such that the set with the currently largest $Q_{MAX}(S)$ is popped out iteratively to calculate its $Q(S)$. While a set S is popped out from SPQ, the child nodes of S are generated on the corresponding set tree and then are inserted into SPQ. Once $Q_{MAX}(S)$ is smaller than the currently largest $Q(S)$ stored, the algorithm terminates and the selection set having the currently largest $Q(S)$ is the optimal solution. Table 10 presents the pseudocode of the *Multi1PNN* algorithm, in which *Quality(S)* calculates $Q(S)$ based on Theorem 3.

7. The details and an example of set tree construction are presented in Appendix H, available in the online supplemental material.

TABLE 11
Notations Introduced in Section 6

$P_{UB}(A)$	upper bound of the probability $P(A)$
A_m	answer A with the m-th largest $P_{UB}(A)$ (the m-th major answer)
$P_k(A, d_{i,j})$	probability that $d_{i,j}$ is the k-th distance in k-PNN answer A
$EQIndex(S, O_i)$	estimated quality improvement after adding O_i to S
$\Delta EQ(O_{i_C}, O_i)$	estimated quality improvement after adding O_i to $\{O_{i_C}\}$

6 Acquisition for k-PNN

In Sections 4 and 5, we have proposed aggregation and pruning strategies for efficient process of 1-PNN. Most of them rely on the examination of the conditional probability for the object with the smallest distance, such that the distances of other objects can be considered as a whole, such as $X_{i,j}$ in Definition 7, which is exploited for deciding $p_{i,j} \times P(\overline{A}_{(i,j)})$, and $E_{S(i,j)}$ in Definition 9, which is used for deciding $P(E_{S(i,j)}) \times P(\overline{A}_{S(i,j)})$. However, deciding the optimal selection for k-PNN becomes intractable, even for 1-acquisition, since an enormous number of k-PNN answer sets are now candidates of the representative answers. Therefore, we devise a heuristic algorithm to solve the data acquisition for k-PNN in reasonable time. The key idea is to identify only a portion of k-PNN answer sets, instead of examining all. Specifically, we focus on only the *major answers*, which are the k-PNN answer sets with sufficiently large probabilities. Afterward, the solution of single-object acquisition will guide an efficient search of s-acquisition.

6.1 Major Answers and 1-Acquisition for k-PNN

Recall that in 1-PNN, for each acquired distance $d_{i,j}$, $p_{i,j} \times P(A_{(i,j)})$ of every answer $A_{(i,j)}$ is necessary to be examined in order to identify $p_{i,j} \times P(\overline{A}_{(i,j)})$ of the representative answer. In k-PNN, however, $A_{(i,j)}$ needs to become an answer set, and the probabilities of C_k^n possible answer sets are required to be derived accordingly. To reduce the computational load, we identify only a portion of k-PNN answer sets as the candidates of the representative answer $\overline{A}_{(i,j)}$. Intuitively, a k-PNN answer set is promising to act as a candidate if the probabilities are sufficiently large conditional on most j. However, the probability of each k-PNN answer set is also unknown. Therefore, we utilize an upper bound of the probability introduced by Cheng et al. [10] to extract the major answers to efficiently process single-object acquisition for k-PNN.

Definition 15. *We define $P_{UB}(A)$ as the upper bound of the probability of a k-PNN answer A, which is the probability that all possible distances of objects in A are not larger than $d^k_{\max}$,*

$$P_{UB}(A) = \prod_{\forall i | O_i \in A} P(d_i \leq d^k_{\max}).$$

To extract the major answers, the algorithm in Section 5.3 is simplified and employed here to generate the

TABLE 12
Major Answers and $MAPTable$ (O_6 Part)

m	A_m	P_{UB}	$P(A_m)$	$d_{6,1}$	$d_{6,2}$	$d_{6,3}$
1	$\{O_2, O_3, O_4\}$	1.00	0.2235	0.0288	0.0631	0.1316
2	$\{O_2, O_3, O_6\}$	0.70	0.0993	0.0881	0.0112	0.0000
3	$\{O_2, O_4, O_6\}$	0.70	0.1194	0.1060	0.0134	0.0000
4	$\{O_3, O_4, O_6\}$	0.70	0.0717	0.0717	0.0000	0.0000
5	$\{O_2, O_3, O_7\}$	0.50	0.0631	0.0168	0.0165	0.0298
6	$\{O_2, O_4, O_7\}$	0.50	0.0635	0.0168	0.0187	0.0280
7	$\{O_3, O_4, O_7\}$	0.50	0.0227	0.0048	0.0072	0.0108
8	$\{O_2, O_6, O_7\}$	0.35	0.0265	0.0232	0.0034	0.0000
9	$\{O_3, O_6, O_7\}$	0.35	0.0112	0.0112	0.0000	0.0000

TABLE 13
Sorted $MAPTable$ (O_6 Part)

$d_{6,1}$		$d_{6,2}$		$d_{6,3}$	
O_2, O_4, O_6	0.1060	O_2, O_3, O_4	0.0631	O_2, O_3, O_4	0.1316
O_2, O_3, O_6	0.0881	O_2, O_4, O_7	0.0187	O_2, O_3, O_7	0.0298
O_3, O_4, O_6	0.0717	O_2, O_3, O_7	0.0165	O_2, O_4, O_7	0.0280
O_2, O_3, O_4	0.0288	O_2, O_4, O_6	0.0134	O_3, O_4, O_7	0.0108
O_2, O_6, O_7	0.0232	O_2, O_3, O_6	0.0112	O_2, O_3, O_6	0.0000
O_2, O_4, O_7	0.0168	O_3, O_4, O_7	0.0072	O_2, O_4, O_6	0.0000
O_2, O_3, O_7	0.0168	O_2, O_6, O_7	0.0034	O_3, O_4, O_6	0.0000
O_3, O_6, O_7	0.0112	O_3, O_4, O_6	0.0000	O_2, O_6, O_7	0.0000
O_3, O_4, O_7	0.0048	O_3, O_6, O_7	0.0000	O_3, O_6, O_7	0.0000

k-PNN answer sets in descending order of P_{UB}. To avoid enumerating all possible k-PNN answer sets, only the first M k-PNN answer sets are extracted as the major answers, and M can be adjusted according to the available computational budget.

Definition 16. *Let A_m denote the major answer with the mth largest P_{UB}.*

After extracting each major answer A_m according to its upper bound, it is necessary to derive the exact probability that A_m is the k-PNN answer. The probability is derived by carefully examining all cases with each object in A_m as the kth object in the answer.

Definition 17. *Let $P_k(A, d_{i,j})$ denote the probability that $d_{i,j}$ is the kth distance in the k-PNN answer A,*

$$P_k(A, d_{i,j}) = p_{i,j} \times \prod_{\forall x | O_x \in A, x \neq i} P(d_x < d_{i,j}) \times \prod_{\forall x | O_x \notin A} P(d_x > d_{i,j}),$$

and the probability of A is

$$P(A) = \sum_{\forall (i,j) | O_i \in A} P_k(A, d_{i,j}).$$

With the above strategy, the following theorem systematically derives $p_{i,j} \times P(A_{(i,j)} = A_m)$.

Theorem 5. *Given a major answer A_m, for $O_i \in A_m$,*

$$p_{i,j} \times P(A_{(i,j)} = A_m) = P_k(A_m, d_{i,j}) + \sum_{\forall x | O_x \in A_m, x \neq i} \sum_{\forall y | d_{x,y} > d_{i,j}} \left(P_k(A_m, d_{x,y}) \times \frac{p_{i,j}}{P(d_i < d_{x,y})} \right);$$

for $O_i \notin A_m$,

$$p_{i,j} \times P(A_{(i,j)} = A_m) = \sum_{\forall x | O_x \in A_m} \sum_{\forall y | d_{x,y} < d_{i,j}} \left(P_k(A_m, d_{x,y}) \times \frac{p_{i,j}}{P(d_i > d_{x,y})} \right).$$

The proof is presented in Appendix E, available in the online supplemental material. If $O_i \in A_m$, Theorem 5 sums up $P_k(A_m, d_{i,j})$ for $d_{i,j}$ itself and $P_k(A_m, d_{x,y})$ with $d_{x,y} > d_{i,j}$ for every other $O_x \in A_m$. On the contrary, if $O_i \notin A_m$, Theorem 5 sums up $P_k(A_m, d_{x,y})$ with $d_{x,y} < d_{i,j}$ for every $O_x \in A_m$. To accumulate $p_{i,j} \times P(A_{(i,j)} = A_m)$ of each (i, j, m), we define the $MAPTable$, denoting *Major Answer Probability Table*, where $MAPTable[i][j][m]$ represents $p_{i,j} \times P(A_{(i,j)} = A_m)$. In each iteration, we consider a major answer A_m with the largest P_{UB} among all k-PNN answers that have not been considered yet. In the m-th iteration, we calculate $P_k(A_m, d_{i,j})$ for each $d_{i,j}$ based on Definition 17, and we update the corresponding entries in the $MAPTable$ following Theorem 5.

After $P_k(A_m, d_{i,j})$ of each $d_{i,j}$ is obtained for $O_i \in A_m$, $p_{i,j} \times P(A_{(i,j)} = A_m)$ of each $d_{i,j}$ is accumulated in $MAPTable[i][j][m]$. When the number of extracted major answers reaches M, we estimate the best 1-acquisition based on the probability in the current $MAPTable$. For each $d_{i,j}$, $p_{i,j} \times P(\overline{A}_{(i,j)})$ is the maximum one of $MAPTable[i][j][m]$ for all m. For each O_i, we sum up $p_{i,j} \times P(\overline{A}_{(i,j)})$ for all j to obtain $Q(\{O_i\})$. Finally, the object with the largest estimated $Q(\{O_i\})$ is the best 1-acquisition. The pseudocode and implementation details of the proposed *SinglekPNN* algorithm introduced as above are in Appendix G, available in the online supplemental material.

Example 3. Considering the data set in Fig. 1, Table 12 presents an example with $k = 3$ and $M = 9$. We find the major answers in descending order of the P_{UB} as shown in Table 12. The rightmost three columns in Table 12 compose the final $MAPTable$ with $i = 6$. According to the $MAPTable$, we estimate $\overline{A}_{(6,1)} = \{O_2, O_4, O_6\}$, $\overline{A}_{(6,2)} = \{O_2, O_3, O_4\}$, and $\overline{A}_{(6,3)}$ is $\{O_2, O_3, O_4\}$. Then we derive $Q(\{O_6\}) = 0.1060 + 0.0631 + 0.1316 = 0.3007$.

6.2 Approach for s-Acquisition for k-PNN

In the following, we extend the algorithm for 1-acquisition for k-PNN to solve s-acquisition by iteratively choosing and adding an object to S. In each iteration, let $EQIndex(S, O_i)$ denote the estimated quality improvement after adding O_i to S, while S is empty initially. For each $O_{i_C} \in S$, let $\Delta EQ(O_{i_C}, O_i)$ denote the estimated quality improvement of selecting object O_i when O_{i_C} is included in S. We estimate $EQIndex(S, O_i)$ according to $\Delta EQ(O_{i_C}, O_i)$.

Definition 18. *For each S,*

$$EQIndex(S, O_i) = \sum_{\forall O_{i_C} \in S} \Delta EQ(O_{i_C}, O_i).$$

To maximize the quality improvement, each iteration of our algorithm selects the object O_i that generates the maximal $EQIndex(S, O_i)$, and then adds O_i to S until S contains s objects. Therefore, $\Delta EQ(O_{i_C}, O_i)$, which is the basic component of $EQIndex(S, O_i)$, plays an important role in estimating the quality improvement. The main idea is to examine the correlation between an existing object in S and a new object to estimate the quality improvement after adding the new object to S, by exploiting the major answers introduced in Section 6.1.

TABLE 14
Notations in Section 7

Notation	Description
LB	Long Beach dataset
MG	Montgomery dataset
-U	Uniform distribution
-G	Gaussian distribution
-l	Number of distance values of an uncertain object
M	Number of major answers

Notation	Description
s	Number of objects for acquisition (s-acquisition)
k	Number of objects for PNN query (k-PNN)
Random	Exact quality of the random 1-acquisition
Estimated	Estimated quality using the proposed approach
Exact	Exact quality using the proposed approach
Partial	An incomplete version of the proposed approach

Definition 19.

$$\Delta EQ(O_{i_C}, O_i) = \sum_{\forall j_C} \left(p_{i_C,j_C} \times \max_{\forall A_m \in I(d_{i_C,j_C}, O_i)} P(A_{(i_C,j_C)} = A_m) \right),$$

where $I(d_{i_C,j_C}, O_i)$ is the union of A_m which meets either of the following conditions.

1. $O_i \in A_m$ *but* $O_i \notin \overline{A}_{(i_C,j_C)}$, *and* $O_{i_C} \notin A_m$ *but* $O_{i_C} \in \overline{A}_{(i_C,j_C)}$.
2. $O_i \in \overline{A}_{(i_C,j_C)}$ *but* $O_i \notin A_m$, *and* $O_{i_C} \notin \overline{A}_{(i_C,j_C)}$ *but* $O_{i_C} \in A_m$.

Theorem 6. *$\Delta EQ(O_{i_C}, O_i)$ is a lower bound on $Q(\{O_{i_C}, O_i\}) - Q(\{O_{i_C}\})$.*

More details of Theorem 6 can be found in Appendix C, available in the online supplemental material. To solve the s-acquisition for k-PNN problem, the proposed *MultikPNN* algorithm first applies the *SinglekPNN* algorithm to identify the major answers stored in the $MAPTable$, and the solution of 1-acquisition is added to S. Then it iteratively selects and adds the object with the largest $EQIndex(S, O_i)$ among all $O_i \notin S$ to S until the size of S reaches the given s. Notice that $MAPTable[i_C][j_C][m]$ is sorted in descending order to find $\Delta EQ(O_{i_C}, O_i)$ efficiently. Implementation details and the pseudocode of the *MultikPNN* algorithm can be found in Appendix G, available in the online supplemental material.

Example 4. Considering the data set in Fig. 1, we present an example with $k = 3$ and $s = 2$. After obtaining the 1-acquisition solution, we first initialize $S = \{O_6\}$. In the first iteration, Table 13lists the sorted $p_{6,j} \times P(A_{(6,j)})$ for each j in $MAPTable$. For $d_{6,1}$, we add $p_{6,1} \times P(A_{(6,1)} = \{O_2, O_3, O_4\})$ to $\Delta EQ(O_6, O_3)$ and $p_{6,1} \times P(A_{(6,1)} = \{O_2, O_4, O_7\})$ to $\Delta EQ(O_6, O_7)$. For $d_{6,2}$, we add $p_{6,2} \times P(A_{(6,2)} = \{O_2, O_4, O_6\})$ to $\Delta EQ(O_6, O_3)$ and $p_{6,2} \times P(A_{(6,2)} = \{O_2, O_3, O_6\})$ to $\Delta EQ(O_6, O_4)$. For $d_{6,3}$, no update is required for each $\Delta EQ(O_6, O_i)$. At the end of the iteration, we find $\Delta EQ(O_6, O_3) = 0.0422$, $\Delta EQ(O_6, O_4) = 0.0112$, $\Delta EQ(O_6, O_7) = 0.0168$. Since $EQIndex$ equals to ΔEQ when $|S| = 1$, we add O_3 to S. The solution is $S = \{O_6, O_3\}$.

7 Experiments

In this section, we conduct extensive experiments on real data sets to evaluate the proposed algorithms for data acquisition. The notations are summarized in Table 14.

7.1 Experimental Setup

Following the experiment setting of previous works on probabilistic nearest-neighbor query [9], [10], we choose LB (Long Beach) data set[8] with $53,144$ intervals distributed in the x-dimension of 10K units, as well as MG (Montgomery) data set containing $39,231$ intervals. Each interval is regarded as an uncertain object and divided into l bins with uniform or Gaussian distribution [9], [10], [11], which is denoted as *-U* or *-G* respectively in the figures. For example, *LB-U-10* is to divide an interval in LB data set into $l = 10$ bins with uniform distribution. The center of each bin is regarded as a possible attribute value of the object with the total probability in the bin as the corresponding probability. Each data point is the average over 100 queries. The query points are chosen uniformly at random from the x-dimension of $10K$ units unless stated otherwise. The default k in experiments is 2. All algorithms are implemented in a PC with an Intel i7 CPU of 2.67 GHz and 3 GB memory. Appendix I, available in the online supplemental material, shows extra experimental results.

7.2 Quality of Data Acquisition

The first part of our experiments is to compare the quality with data acquisition and the quality without data acquisition. Fig. 3 shows the quality of s-acquisition for 1-PNN over different s. For LB data set in Fig. 3a, as s increases from 1 to 6, data acquisition boosts the quality from 0.29 to 0.63 for the uniform distribution, and it improves the quality from 0.43 to 0.8 for Gaussian distribution. Data acquisition leads to a more informative result for Gaussian distribution because the probabilities of representative answers tend to vary more significantly, and thus it is easier to generate representative answers with a large probability. The quality of MG data set is slightly larger since it has a smaller number of objects. Fig. 3b shows that the quality increases for a smaller l. In this case, each distance is associated with a larger probability, and $P(\overline{A})$ of representative answer $\overline{A}$ tends to enjoy a more dominating probability.

Figs. 4a and 4b present the quality of 1-acquisition for k-PNN in LB data set, where the quality ratio is the quality with data acquisition over the quality without data acquisition. Fig. 4a shows the exact quality of different k with 20 and 100 major answers defined in Section 6, where *Random* presents the exact quality of randomly picking up an object from the candidates. The result indicates that the quality of the proposed algorithm

8. Available at http://www.census.gov/geo/www/tiger/ or http://www.rtreeportal.org/.

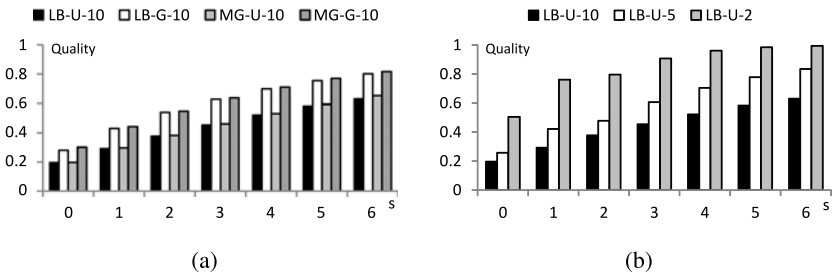

Fig. 3. Quality of s-acquisition for 1-PNN.

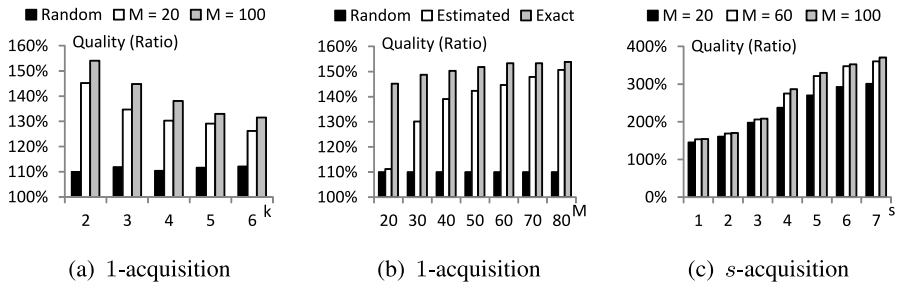

Fig. 4. Quality of data acquisition for k-PNN.

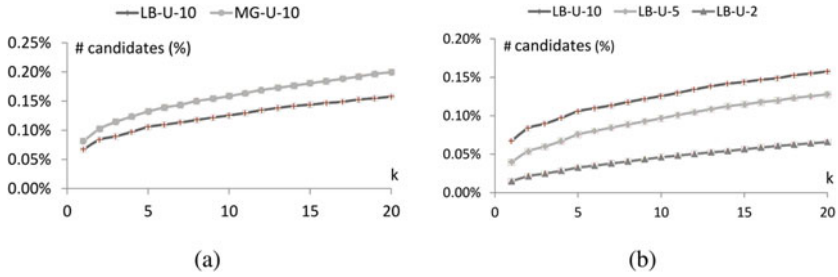

Fig. 5. Number of candidate objects.

significantly outperforms *Random* even with $M = 20$, and more major answers can further improve the quality. When $M = 100$, over 30 percent performance improvement can be achieved even with single-object acquisition in k-PNN. Nevertheless, the results also manifest that more major answers are suggested to be extracted for a larger k, since the number of new answers grows in this case. Fig. 4b shows the quality over different numbers of major answers. While *Exact* and *Random* represent the exact quality of the proposed approach and the exact quality of randomly picking up an object, respectively, *Estimated* presents the estimated quality based on major answers. When the number of major answers increases, the estimated quality first significantly increases and then converges to the exact quality. In contrast, the increment of the exact quality is marginal, indicating that a reasonable number of major answers is sufficient for data acquisition to achieve satisfying exact quality. Fig. 4c compares the exact quality of s-acquisition for k-PNN for 20, 60, or 100 major answers. Similarly, more major answers generate larger quality, and the quality also improves as s increases. Moreover, the improvement of a larger s becomes more significant with more major answers because more pairwise relationships among objects can be ascertained.

7.3 Efficiency of Data Acquisition

The second part of our experiments is to evaluate the efficiency of the proposed approaches. Fig. 5 first finds the percentage of the objects remained after the candidate identification process in Section 3.2. It is impressive that most objects in both real data sets can be effectively pruned. For both LB and MG data sets, Fig. 5a shows that the number of candidates increases with a larger k because $d^k_{\max}$ grows in this case, and more objects tend to stay in the candidate region. Fig. 5b shows that fewer candidates are generated for a smaller l because it becomes more difficult for any two objects to overlap in this case.

Figs. 6a and 6b evaluate the computation time of the proposed approach for s-acquisition of 1-PNN. Fig. 6a indicates that a larger l requires more computation time. When $s = 3$, the brute-force approach consumes about 500 seconds, while the proposed algorithm only requires about 5 milliseconds. Moreover, the computation time of the proposed approach increases much more smoothly than the computation time of the brute-force approach does, thanks to various strategies developed to simplify the probability derivation and to prune off unnecessary objects. Besides, since it is computation intractable to conduct the brute-force approach on $l = 10$, Fig. 6b here compares the number of selection sets S involved in the computation. With the set

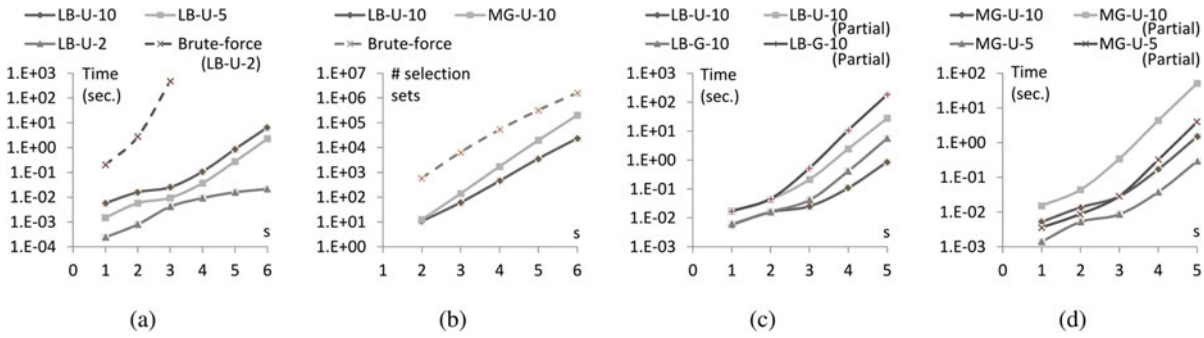

Fig. 6. Computation time of s-acquisition for 1-PNN.

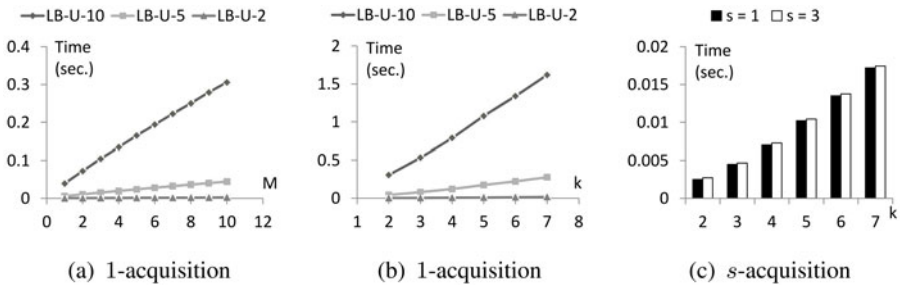

Fig. 7. Computation time of data acquisition for k-PNN.

quality estimation in Section 5.2, the number of possible selection sets to be examined in the proposed algorithm is much smaller than in the brute-force approach. Moreover, $Q_{MAX}(S)$ in the uniform distribution is tighter in this case and thus able to prune more unnecessary objects because the difference of the Q_B in varied objects becomes smaller.

Besides, to demonstrate the efficiency of the proposed approach for s-acquisition of 1-PNN, we implement an incomplete version of the proposed approach, denoting as *Partial*. The incomplete version does not include the sequential derivations of $P(A_{(i,j)} = \{O_x\})$ and $P(A_{S(i,j)} = \{O_x\})$ in Lemmas 8 and 15. Figs. 6c and 6d manifest that *Partial* incurs much more time than the proposed approaches, for any data set, any l value, and any distribution. The difference of computation time significantly increases even in log scale, demonstrating that the proposed sequential probability derivations enable us to efficiently solve the data acquisition for 1-PNN problem. Besides, in line with the trend in Fig. 6b, the results with Gaussian distribution require more time than the results with uniform distribution. For the data with Gaussian distribution, objects are inclined to have larger 1-acquisition quality and thereby generate a large $Q_{MAX}(S)$, such that more possible selection sets are necessary to be examined in detail.

Fig. 7a compares the computation time of the proposed algorithm for 1-acquisition of k-PNN over different numbers of major answers, while Fig. 7b compares the computation time over different k with 10 major answers. In the figures, the computation time is proportional to k and the number of major answers. Besides, the time grows with a larger l. With the proposed algorithm, the computation time of s-acquisition for k-PNN is very close to the computation time of 1-acquisition for k-PNN, as shown in Fig. 7c. The above results demonstrate that in the algorithm for s-acquisition, most time is consumed for 1-acquisition, such that its major answers are able to be properly exploited to sequentially select each of the s objects afterward.

8 Conclusion

Different from the previous works that directly handle uncertain data for query processing, this paper proposes data acquisition to improve the quality of the result for k-PNN. We formulate a new optimization problem, s-acquisition for k-PNN, and prove that every uncertain k-PNN result can be improved by selecting proper data objects for acquisition. To efficiently process data acquisition, we propose the candidate identification process to trim the objects that cannot improve the k-PNN results. Afterwards, we devise an algorithm to find the optimal solution of data acquisition for 1-PNN with various strategies such as *NAPTable* and an upper bound Q_{MAX}. For k-PNN, we propose a heuristic algorithm based on the notion of major answers for quality estimation. Extensive experiments for evaluating the efficiency and effectiveness of the proposed algorithms are performed, and the results demonstrate that the quality of the k-PNN result can be effectively improved with only a limited number of objects with data acquisition.

References

[1] S. Abiteboul, P.C. Kanellakis, and G. Grahne, "On the Representation and Querying of Sets of Possible Worlds," *Proc. ACM SIGMOD Int'l Conf. Management of Data (SIGMOD)*, 1987.

[2] P.K. Agarwal, A. Efrat, S. Sankararaman, and W. Zhang, "Nearest-Neighbor Searching under Uncertainty," *Proc. 31st ACM Symp. Principles of Database Systems (PODS)*, 2012.

[3] C.C. Aggarwal and P.S. Yu, "A Framework for Clustering Uncertain Data Streams," *Proc. 24th IEEE Int'l Conf. Data Eng. (ICDE)*, 2008.

[4] R. Cheng, D.V. Kalashnikov, and S. Prabhakar, "Evaluating Probabilistic Queries over Imprecise Data," *Proc. ACM SIGMOD Int'l Conf. Management of Data (SIGMOD)*, 2003.

[5] R. Akbarinia, P. Valduriez, and G. Verger, "Efficient Evaluation of SUM Queries over Probabilistic Data," *IEEE Trans. Knowledge and Data Eng.*, vol. 25, no. 4, Apr. 2013.
[6] G. Beskales, M.A. Soliman, and I.F. Ilyas, "Efficient Search for the Top-k Probable Nearest Neighbors in Uncertain Databases," *Proc. 34th Int'l Conf. Very Large Data Bases (VLDB)*, 2008.
[7] C. Bohm, F. Fiedler, A. Oswald, C. Plant, and B. Wackersreuther, "Probabilistic Skyline Queries," *Proc. 18th ACM Conf. Information and Knowledge Management (CIKM)*, 2009.
[8] J. Chen and R. Cheng, "Quality-Aware Probing of Uncertain Data with Resource Constraints," *Proc. 20th Int'l Conf. Scientific and Statistical Database Management (SSDBM)*, 2008.
[9] J. Chen, R. Cheng, M.F. Mokbel, and C.-Y. Chow, "Scalable Processing of Snapshot and Continuous Nearest-Neighbor Queries over One-Dimensional Uncertain Data," *Proc. 35th Int'l Conf. Very Large Data Bases (VLDB)*, 2009.
[10] R. Cheng, L. Chen, J. Chen, and X. Xie, "Evaluating Probability Threshold k-Nearest-Neighbor Queries over Uncertain Data," *Proc. 12th Int'l Conf. Extending Database Technology (EDBT)*, 2009.
[11] R. Cheng, J. Chen, and X. Xie, "Cleaning Uncertain Data with Quality Guarantees," *Proc. 34th Int'l Conf. Very Large Data Bases (VLDB)*, 2008.
[12] R. Cheng, E. Lo, X.S. Yang, M.-H. Luk, X. Li, and X. Xie, "Explore or Exploit? Effective Strategies for Disambiguating Large Databases," *Proc. 36th Int'l Conf. Very Large Data Bases (VLDB)*, 2010.
[13] R. Cheng, D.V. Kalashnikov, and S. Prabhakar, "Querying Imprecise Data in Moving Object Environments," *IEEE Trans. Knowledge and Data Eng.*, vol. 16, no. 9, pp. 1112-1127, Sept. 2004.
[14] R. Cheng, Y. Xia, S. Prabhakar, R. Shah, and J.S. Vitter, "Efficient Indexing Methods for Probabilistic Threshold Queries over Uncertain Data," *Proc. 30th Int'l Conf. Very Large Data Bases (VLDB)*, 2004.
[15] J. Dougherty, R. Kohavi, and M. Sahami, "Supervised and Unsupervised Discretization of Continuous Features," *Proc. 12th Int'l Conf. Machine Learning (ICML)*, 1995.
[16] N. Fuhr and T. Rolleke, "A Probabilistic Relational Algebra for the Integration of Information Retrieval and Database Systems," *ACM Trans. Information Systems*, vol. 15, pp. 32-66, 1997.
[17] T. Ge and S.B. Zdonik, "Handling Uncertain Data in Array Database Systems," *Proc. 24th IEEE Int'l Conf. Data Eng. (ICDE)*, 2008.
[18] S. Gunnemann, H. Kremer, and T. Seidl, "Subspace Clustering for Uncertain Data," *Proc. 10th SIAM Int'l Conf. Data Mining (SDM)*, 2010.
[19] M.E. Khalefa, M.F. Mokbel, and J.J. Levandoski, "Skyline Query Processing for Uncertain Data," *Proc. 19th ACM Conf. Information and Knowledge Management (CIKM)*, 2010.
[20] H.-P. Kriegel, P. Kunath, and M. Renz, "Probabilistic Nearest-Neighbor Query on Uncertain Objects," *Proc. 12th Int'l Conf. Database Systems for Advanced Applications (DASFAA)*, 2007.
[21] J. Li and A. Deshpande, "Ranking Continuous Probabilistic Datasets," *Proc. 36th Int'l Conf. Very Large Data Bases (VLDB)*, 2010.
[22] X. Lian and L. Chen, "Efficient Processing of Probabilistic Reverse Nearest Neighbor Queries over Uncertain Data," *Int'l J. Very Large Data Bases*, vol. 18, no. 3, pp. 787-808, June 2009.
[23] X. Liu, M. Ye, J. Xu, Y. Tian, and W.-C. Lee, "K-Selection Query over Uncertain Data," *Proc. 15th Int'l Conf. Database Systems for Advanced Applications (DASFAA)*, 2010.
[24] L.V.S. Lakshmanan, N. Leone, R. Ross, and V.S. Subrahmanian, "ProbView: A Flexible Probabilistic Database System," *ACM Trans. Database Systems (TODS)*, vol. 22, no. 3, pp. 419-469, Sept. 1997.
[25] C. Olston, J. Jiang, and J. Widom, "Adaptive Filters for Continuous Queries over Distributed Data Streams," *Proc. ACM SIGMOD Int'l Conf. Management of Data (SIGMOD)*, 2003.
[26] M. Potamias, F. Bonchi, A. Gionis, and G. Kollios, "K-Nearest Neighbors in Uncertain Graphs," *Proc. 36th Int'l Conf. Very Large Data Bases (VLDB)*, 2010.
[27] S. Singh, C. Mayfield, R. Shah, S. Prabhakar, S.E. Hambrusch, J. Neville, and R. Cheng, "Database Support for Probabilistic Attributes and Tuples," *Proc. 24th IEEE Int'l Conf. Data Eng. (ICDE)*, 2008.
[28] M.A. Soliman, I.F. Ilyas, and K.C.-C. Chang, "Top-k Query Processing in Uncertain Databases," *Proc. 23th IEEE Int'l Conf. Data Eng. (ICDE)*, 2007.
[29] B. Yang, H. Lu, and C.S. Jensen, "Probabilistic Threshold K Nearest Neighbor Queries over Moving Objects in Symbolic Indoor Space," *Proc. 13th Int'l Conf. Extending Database Technology (EDBT)*, 2010.
[30] S.M. Yuen, Y. Tao, X. Xiao, J. Pei, and D. Zhang, "Superseding Nearest Neighbor Search on Uncertain Spatial Databases," *IEEE Trans. Knowledge and Data Eng.*, vol. 22, no. 7, pp. 1041-1055, July 2010.
[31] Y. Zhang, X. Lin, G. Zhu, W. Zhang, and Q. Lin, "Efficient Rank Based KNN Query Processing over Uncertain Data," *Proc. 26th IEEE Int'l Conf. Data Eng. (ICDE)*, 2010.
[32] K. Zheng, Z. Huang, A. Zhou, and X. Zhou, "Discovering the Most Influential Sites over Uncertain Data: A Rank-Based Approach," *IEEE Trans. Knowledge and Data Eng.*, vol. 24, no. 12, pp. 2156-2169, Dec. 2012.
[33] Z. Zou, J. Li, H. Gao, and S. Zhang, "Frequent Subgraph Pattern Mining on Uncertain Graph Data," *Proc. 18th ACM Conf. Information and Knowledge Management (CIKM)*, 2009.
[34] M. Azizyan, I. Constandache, and R. Roy Choudhury, "SurroundSense: Mobile Phone Localization via Ambience Fingerprinting," *Proc. MobiCom*, 2009.
[35] P. Bahl and V. Padmanabhan, "RADAR: An In-Building RF-Based User Location and Tracking System," *Proc. IEEE INFOCOM*, 2000.
[36] A. Goel, S. Guha, and K. Munagala, "Asking the Right Questions: Model-Driven Optimization Using Probes," *Proc. 25th ACM Symp. Principles of Database Systems (PODS)*, 2006.
[37] A. Goel, S. Guha, and K. Munagala, "How to Probe for an Extreme Value," *ACM Trans. Algorithms*, vol. 7, no. 1, article 12, Nov. 2010.
[38] M.A. Soliman and I.F. Ilyas, "Ranking with Uncertain Scores," *Proc. 25th IEEE Int'l Conf. Data Eng. (ICDE)*, 2009.
[39] W. Elmenreich, "Fusion of Continuous-valued Sensor Measurements using Confidence-Weighted Averaging," *J. Vibration and Control (JVC)*, vol. 13, pp. 1303-1312, Sept. 2007.
[40] M. Hua, J. Pei, W. Zhang, and X. Lin, "Ranking Queries on Uncertain Data: A Probabilistic Threshold Approach," *Proc. ACM SIGMOD Int'l Conf. Management of Data (SIGMOD)*, 2008.

Yu-Chieh Lin received the BS and PhD degrees from the Department of Electrical Engineering, National Taiwan University, in 2005 and 2013, respectively. She is currently a postdoctoral fellow at the Research Center of Information Technology Innovation (CITI) in the Academia Sinica, Taiwan. Her research interests include data mining and social networks.

De-Nian Yang received the PhD degree from National Taiwan University, in 2004. His research interests include social networks, mobile data management, and mobile multimedia networking. His research topics for social networks include social influence, viral marketing, query processing, and privacy preserving, and the research results have been published in renowned international conferences and journals, such as in *VLDB*, *KDD*, *ICDM*, *CIKM*, *PAKDD*, *MDM*, *DASFAA*, *TKDE*, *and TMC*. His research results have also been featured by MIT Technology Review and ACM TechNews. He received NSC Project for Excellent Junior Research Investigators, Career Development Award in Academia Sinica, Emerging Technologies Prize in SIGGRAPH Asia, K.T. Li Distinguished Young Scholar Award in ACM Taipei/Taiwan Chapter, Exploration Research Award in Pan Wen Yuan Foundation, and Best Student Paper Award in IEEE ICME. He is a senior member of IEEE and a member of ACM.

Hong-Han Shuai received the BS degree in electrical engineering from National Taiwan University in 2007. He is currently working toward the PhD degree at the Graduate Institute of Communication Engineering, National Taiwan University, Taiwan. His research interests include data mining and social networks.

Ming-Syan Chen received the BS degree in electrical engineering from National Taiwan University, Taipei, Taiwan, and the MS and PhD degrees in computer, information and control engineering from The University of Michigan, Ann Arbor, MI, in 1985 and 1988, respectively. He is now a distinguished research fellow and the director of Research Center of Information Technology Innovation (CITI) in the Academia Sinica, Taiwan, and is also a distinguished professor jointly appointed by EE Department, CSIE Department, and Graduate Institute of Communication Engineering (GICE) at National Taiwan University. He was a research staff member at IBM Thomas J. Watson Research Center, Yorktown Heights, NY, from 1988 to 1996, the director of GICE from 2003 to 2006, and also the president/CEO of Institute for Information Industry (III), which is one of the largest organizations for information technology in Taiwan, from 2007 to 2008. His research interests include databases, data mining, social networks, and multimedia networking, and he has published more than 300 papers in his research areas. In addition, he received numerous awards for his research, teaching, inventions and patent applications. He is a fellow of the IEEE and ACM.