APPENDIX A
EQUIVALENCE OF $\text{cost}_{\text{SumMax2}}$ AND $\text{cost}_{\text{Sum}}$

We have the following lemma to show the functionality of $\text{cost}_{\text{SumMax2}}$ and $\text{cost}_{\text{Sum}}$ are equivalent.

Lemma 4. Let $S$ be an object set. $\text{cost}_{\text{SumMax2}}(S) = \text{cost}_{\text{Sum}}(S)$.

Proof. Let $(o_1, o_2) = \text{arg max}_{o_1, o_2 \in S} d(o_1, o_2)$. We have

$$\text{cost}_{\text{SumMax2}}(S) = \max\{\sum_{o \in S} d(o, q_1), d(o_1, o_2)\}.$$  

Note that $d(o_1, q_1) \geq d(o_1, o_2)$ by triangle inequality and

$$\sum_{o \in S} d(o, q_1) \geq \sum_{o \in S} d(o, q_2).$$

Thus, $\text{cost}_{\text{SumMax2}}(S) = \sum_{o \in S} d(o, q_1) = \text{cost}_{\text{Sum}}(S)$. □

This lemma suggests that it is sufficient to consider one of these two cost functions. In this paper, we focus the discussion on $\text{cost}_{\text{Sum}}$.

APPENDIX B
PROOF OF THEOREM 1

We first give the decision problem of CoSKQ. Given a set $O$ of spatial objects each $o \in O$ associated with a location $o.\lambda$ and a set of keywords $o.\psi$, a query $q$ consisting of a query location $q.\lambda$ and a set of query keywords $q.\psi$, and a real number $C$, the problem is to determine whether there exists a set $S$ of objects in $O$ such that $S$ covers the query keywords and $\text{cost}_{\text{unified}}(S)$ is at most $C$.

We then prove by transforming the 3-satisfiability (3-SAT) problem which is known to be NP-Complete to the CoSKQ problem and showing the equivalence between two problems. The description of the 3-SAT problem is given as follows. Let $U$ be a set of literals $\{e_1, e_2, ..., e_n, \overline{e}_m\}$ where $\overline{e}_m$ is the negation of $e_m$. Given an expression $E = C_1 \land C_2 \land ... \land C_m$ where $C_j = \neg e_j \vee y_j \vee z_j$ and $x_j, y_j, z_j \in U$ for $1 \leq j \leq m$, the problem is to determine whether there exists a truth assignment for $e_i$ for $1 \leq i \leq n$ such that $E$ is true.

Based on the value of parameter $\phi_1$, we use different transformations.

Case 1. $\phi_1 = 1$. We construct a set $O$ of $2n$ objects as follows. For each literal $e_i$ in $U$, we create an object $o_i$ in $O$, and for each literal $\overline{e}_i$ in $U$, we create an object $\overline{o}_i$ in $O$. In total, there are $2n$ objects in $O$. We set the locations of the objects in $O$ such that they are all located at the same place i.e., for any $o \in O$, $o.\lambda$ is identical.

Besides, for each object $o_i (1 \leq i \leq n)$, we set $o_i.\psi$ such that $o_i.\psi$ includes a keyword $k_i$ corresponding to $e_i$ and a keyword $k_i'$ corresponding to $\overline{e}_i$. We set $\overline{o}_i.\psi$ includes $k_i$ and all $k_j'$'s with $C_j$ involving $\overline{e}_j$ for $1 \leq j \leq m$. We construct a query $q$ by setting $q.\lambda$ arbitrarily and $q.\psi$ to be a set of $m + n$ keywords, $\{k_1, k_2, ..., k_n, k_1', k_2', ..., k_m\}$. The above transformation process could be done in polynomial time. We consider the following subcases for setting $C$.

Case 2(a). $\phi_2 = 1$. We set $C = 3 - \epsilon$ where $\epsilon$ is close to zero.

Case 2(b). $\phi_2 = \infty$. We set $C = 2 - \epsilon$ where $\epsilon$ is close to zero.

We show the equivalence between two problem instances as follows. Suppose that the answer of the 3-SAT problem is “yes”, i.e., there exists a truth assignment for the literals in $U$ such that $E$ is correct. We denote the truth assignment by a set $T$ of literals which are true under the assignment. Note that $T$ has exactly $n$ literals and $e_i$ and $\overline{e}_i$ do not appear in $T$ simultaneously for any $1 \leq i \leq n$. Then, it could be verified that the set of objects each corresponding to a literal in $T$ covers $q.\psi$ and the cost of the set at most $C$, and thus the answer of the CoSKQ problem is also “yes”. Suppose that the answer of the CoSKQ problem is “yes”. Let $S$ be the set of objects in $O$ that covers $q.\psi$ and has the cost at most $C$. We know that object $o_i$ and $\overline{o}_i$ are not included in $S$ simultaneously. It could be verified that with the truth assignment represented by the set of literals corresponding to the objects in $S$, $E$ is correct, and thus the answer of the 3-SAT problem is also “yes”. □

APPENDIX C
PRUNING BASED ON DOMINANCE

To improve the efficiency of the algorithm, we propose a pruning strategy to prune the search space when $\alpha = 1$ and $\phi_1 = 1$. Before we give the strategy, we first introduce the concept of dominance. Given a query $q$, two objects $o_1$ and $o_2$, we say $o_1$ dominate $o_2$ if the following two conditions are satisfied. (1) $d(o_1, q) < d(o_2, q)$, and (2) all keywords in $q.\psi$ that are covered by $o_2$ can be covered by $o_1$, i.e. $q.\psi \cap o_1.\psi \supseteq q.\psi \cap o_2.\psi$. A dominant object is defined to be an object that is not dominated by any other objects.

Then we have the following lemma to prune the objects that are not dominant objects.

Lemma 5. When $\alpha = 1$ and $\phi_1 = 1$, all objects in the optimal solution $S$ are dominant objects. □

Proof: We prove this by contradiction. Let an object $o \in S$ that is not a dominant object. Then, there must exist an object $o'$ that dominate $o$. Note that $o'$ also covers the query keywords covered by $o$ and is closer to $q$. We can construct a better solution $S' = S \setminus \{o\} \cup \{o'\}$, which contradicts the fact that $S$ is the optimal solution.

Based on this lemma, it is sufficient for the algorithm to consider the dominant objects only when enumerating the object sets. Specifically, whenever the algorithm performs a range query, it discards the objects that are being dominated and proceeds with the dominant objects.

APPENDIX D
BETTER IMPLEMENTATION BASED ON INFORMATION RE-USE

To implement the Unified-A algorithm efficiently, we have the following implementation strategies. First, when the algorithm finding
the set of all relevant objects in $R_o$ (line 5 in Algorithm 4), instead of issuing a range query in each iteration, it re-uses the information from the previous iteration by maintaining the region $R_o$ dynamically. Specifically, consider one iteration. The algorithm finds a feasible set that has an object $o$ as a key query-object distance contributor in the region $R_o$. After it finishes the current iteration, it adds $o$ into $R_o$ (when $\phi_1 \in [1, \infty)$), or removes $o$ from $R_o$ (when $\phi_1 = -\infty$).

Second, when the algorithm performs the iterative process (lines 8-14 in Algorithm 4), instead of searching for the object with minimum ratio (distance) from $\mathcal{O}'$ in each iteration, it maintains a heap structure for storing the objects. Specifically, when $\phi_1 = 1$, the key of the objects in the heap are the ratios, and the heap is updated after each object is picked. When $\phi_1 \in \{-\infty, -\infty\}$, the key of the objects in the heap are the distances, and in each iteration the algorithm picks the relevant object with the smallest distance.

APPENDIX E

EXPERIMENTAL RESULTS ON THE DATASETS GN AND WEB

In the following, we present the experimental results on the datasets GN and Web of varying $|q, \psi|$. Following the existing studies [1], [17], we vary the number of query keywords (i.e., $|q, \psi|$) from $\{3, 6, 9, 12, 15\}$.

(1) $cost_{MinMax}$. The results for $cost_{MinMax}$ on the datasets GN and Web are shown in Figure 18 and Figure 19 respectively, which are similar to that on the dataset Hotel. The result of running time of Cao-E1 for $|q, \psi| = 15$ is not shown in Figure 19 simply because it ran for more than 10 hours (this applies for all the following results).

(2) $cost_{MinMax}$. The results for $cost_{MinMax}$ on the datasets GN and Web are shown in Figure 20 and Figure 21 respectively, which are similar to those for $cost_{MinMax}$.

(3) $cost_{Sum}$. The results for $cost_{Sum}$ on the datasets GN and Web are shown in Figure 22 and Figure 23 respectively. According to the results, Unified-E runs slower than Cao-E2 but still within a reasonable time (e.g. within 10 seconds on the largest dataset Web). Besides, Unified-A has a very similar running time as Cao-A3, while Unified-A can always obtain an approximation ratios of 1.

(4) $cost_{SumMax}$. The results for $cost_{SumMax}$ on the datasets GN and Web are shown in Figure 24 and Figure 25 respectively, which are similar to that on the dataset Hotel.

(5) $cost_{MaxMax}$. The results for $cost_{MaxMax}$ on the datasets GN and Web are shown in Figure 26 and Figure 27 respectively, which are similar to that on the dataset Hotel.

(6) $cost_{MaxMax2}$. The results for $cost_{MaxMax2}$ on the datasets GN and Web are shown in Figure 28 and Figure 29 respectively, which are similar to that on the dataset Hotel.

(7) $cost_{Max}$. The results for $cost_{Max}$ on the datasets GN and Web are shown in Figure 30 which is similar to that on the dataset Hotel. According to the results, both Unified-E and Unified-A run very fast, e.g. they ran less than 6 ms for all settings of $|q, \psi|$. 
APPENDIX

F

SCALABILITY TEST

(2) \( \text{cost}_{\text{MinMax}} \). The results for \( \text{cost}_{\text{MinMax}} \) are shown in Figure 37. According to Figure 37(a), Unified-E is faster and more scalable than Cao-E1, e.g., on a dataset with 6M objects, Unified-E ran for a couple of seconds while Cao-E1 ran for more than 10 hours. Besides, similar to the case of \( \text{cost}_{\text{MinMax}} \), Unified-A runs slightly slower than Cao-A1, but gives much better approximation ratio, e.g. the median of approximation ratios of Unified-A are 1 on all settings while that of Cao-A1 are larger than 1.

(3) \( \text{cost}_{\text{Sum}} \). The results for \( \text{cost}_{\text{Sum}} \) are shown in Figure 38. According to Figure 38(a), Unified-E is very scalable when the number of objects is large, e.g., it ran slightly longer than 1 second on a dataset with 10M objects. Besides, we noticed that Cao-E2 has a very good performance and it even runs as fast as the approximation algorithms. The reason could be as follows. With the number of objects grows, the number of relevant objects

---

**Fig. 24: Effect of \(|q.\psi|\) on \(\text{cost}_{\text{SumMax}}\) (GN)**

**Fig. 25: Effect of \(|q.\psi|\) on \(\text{cost}_{\text{SumMax}}\) (Web)**

**Fig. 26: Effect of \(|q.\psi|\) on \(\text{cost}_{\text{MaxMax}}\) (GN)**

**Fig. 27: Effect of \(|q.\psi|\) on \(\text{cost}_{\text{MaxMax}}\) (Web)**

**Fig. 28: Effect of \(|q.\psi|\) on \(\text{cost}_{\text{MaxMax2}}\) (GN)**

**Fig. 29: Effect of \(|q.\psi|\) on \(\text{cost}_{\text{MaxMax2}}\) (Web)**

**Fig. 30: Effect of \(|q.\psi|\) on \(\text{cost}_{\text{Max}}\)**
becomes large. Both approximate algorithms have to re-compute the ratio for the remaining nodes in the heap and re-organize the heap after picking each object, whose cost becomes expensive when the number of relevant objects is large. In contrast, Cao-E2 maintains a heap structure though, it does not have to re-examine the nodes after processing a node. Unified-A has similar running times as Cao-A3 but gives better approximation ratios than Cao-A3 (Figure 32(b)). Specifically, Unified-A can achieve near-to-optimal approximation ratios on all setting while Cao-A3 has its largest approximation ratios up to 1.279.

4) $\text{cost}_{\text{Sum}}$. Same as the experiments of varying $|o,\psi|$ for $\text{cost}_{\text{Sum}}$, we used the setting of $|q,\psi| = 8$ for the scalability test experiments for $\text{cost}_{\text{Sum}}$. Particularly, the results for $\text{cost}_{\text{Sum}}$ are shown in Figure 33. According to Figure 33(a), Unified-E and Cao-E1 have similar running times and Unified-A and Cao-A3 also have similar running times, but Unified-A gives a better approximation ratio than Cao-A3 (Figure 33(b)).

5) $\text{cost}_{\text{MaxMax}}$. The results for $\text{cost}_{\text{MaxMax}}$ are shown in Figure 34. According to Figure 34(a), Unified-E runs similarly fast as Long-E, and both of them run faster than Cao-E1. According to Figure 34(b) and (c), Unified-A has similar running times with Cao-A2, both of them run faster than Long-A and slower than Cao-A1, and Unified-A is one of the two algorithms (the other is Long-A which runs slower than Unified-A) which give the best approximation ratios. Specifically, the largest approximation ratios of Unified-A is only 1.109, which is small, while that of Cao-A1 and Cao-A2 are 1.928 and 1.524, respectively.

6) $\text{cost}_{\text{MaxMax2}}$. The results for $\text{cost}_{\text{MaxMax2}}$ are shown in Figure 35. According to Figure 35(a), Unified-E runs similarly fast as Long-E, and both of them run faster than Cao-E1. According to Figure 35(b) and (c), Unified-A has similar running times with Cao-A2, both of them run faster than Long-A and slower than Cao-A1, and Unified-A is one of the two algorithms (the other is Long-A which runs slower than Unified-A) which give the best approximation ratios. Specifically, the largest approximation ratios of Unified-A is only 1.109, which is small, while that of Cao-A1 and Cao-A2 are 2.456 and 1.345, respectively.

7) $\text{cost}_{\text{Max}}$. The results for $\text{cost}_{\text{Max}}$ are shown in Figure 36. According to the results, both Unified-E and Unified-A runs very fast, e.g. they ran within 1 second on a dataset with 10M objects.