報告番号	※甲　　　　第　　　　　号

主　論　文　の　要　旨

論文題目　Combining Multiple Interests in Decision Support Queries
（複数の興味の組合せによる意思決定支援問合せに関する研究）

氏　　名　GUO Xi（郭　茜）

論 文 内 容 の 要 旨

A decision support query is used to recommend handful appropriate objects to a user to help the user's decisions. The user specifies multiple interests as a query and the system outputs the objects that may be interesting to the user as the answer. The user can choose the favorite ones from among the recommended objects. Typically, there are two types of queries to recommend objects: top-*k* queries and skyline queries.

Top-*k* queries retrieve objects based on their scores. The scores are used to evaluate the usefulness of the objects. The scoring function is defined by the user by combining her multiple interests. Sorting the objects by their scores, the system outputs the *k* objects at the top of the ranking list as the answer. However, asking users to define scoring functions is not reasonable because users may not be experts for defining the functions. This is one of the main disadvantages of top-*k* queries.

To overcome this disadvantage, skyline queries were proposed. The skyline queries can recommend objects without relying on any scoring functions. They use the notion of dominance; an object dominates another object if it is better in at least one attribute and not worse in all the other attributes. Given a set of objects with multiple attributes, an object would not be recommended if it is dominated by some other objects.

Generally speaking, the studies in this thesis fall into the skyline query category. In this thesis, two types of queries are proposed and investigated: direction-based surrounder (DBS) queries for spatial datasets and combination skyline queries for multi-attribute datasets.

A DBS query can be applied to the location-based services that recommend spatial objects to users. In the conventional location-based services, the most popular recommendation method is to select the nearest objects of the user. For spatial objects, however, not only their distances but also their directions are important. Motivated by this idea, a DBS query retrieves the nearest objects around the user from all the different directions.

In this thesis, two types of DBS queries are defined in terms of the two-dimensional Euclidean space and road networks. In the Euclidean space, we consider two objects a and b are in the same direction with respect to the user's position q if their included angle $\angle aqb$ is bounded by a threshold specified by the user at the query time. In a road network, we consider two objects a and b are in the same direction if their shortest paths to q overlap. We say object a dominates object b if they are in the same direction and meanwhile a is closer to q than b. All the objects that are not dominated by others based on this dominance relationship constitute the DBSs. The non-dominated objects found in the Euclidean space are called E-DBSs, while the ones found in a road network are called R-DBSs.

In the thesis, DBS queries are studied in both snapshot and continuous settings, and extensive experiments are performed using both real and synthetic datasets to evaluate the proposed algorithms. A snapshot query finds the DBSs according to the current position of a user. A continuous query updates the DBSs when the user is moving. In order to answer snapshot queries in Euclidean spaces, two properties are observed to reduce the search space. For snapshot queries in road networks, we calculate the shortest path of every object and then determine whether it is dominated or not. To answer continuous queries is more difficult than to answer snapshot queries because both the distances and the directions are changing due to the movement of the user. The basic idea of the algorithm is that we update the DBSs only at the change moment rather than updating DBSs at every moment. The change moments can be predicated when the user starts to move. The experimental results demonstrate that the proposed algorithms can answer DBS queries efficiently.

The second study is for the combination skyline queries. A combination skyline query retrieves fixed-size combinations of objects that are not dominated by any other possible combinations. These selected combinations are called the skyline combinations. Combination skyline queries can be applied to many application scenarios such as selecting desired portfolios consisting of stocks. To answer such queries is technically challenging because the traditional skyline approaches only work well when objects are handled individually. In other words, they cannot handle an exponential number of combinations efficiently.

From the observations, it is shown that the number of skyline combinations is far smaller than that of the combinations that can be enumerated. The proposed pattern-based pruning (PBP) algorithm can retrieve the skyline combinations without enumerating and checking all of the combinations. Using an R-tree for indexing objects, the algorithm can prune the candidates efficiently. The solution is based on object-selecting patterns that indicate the number of objects to be selected from each minimum bounding rectangle (MBR) in the R-tree. Two major pruning conditions are proposed to avoid unnecessary expansions and enumerations, as well as a technique to reduce space consumption on storing the skyline for each rule in the object-selecting pattern. The efficiency of the proposed algorithms is demonstrated by the extensive experiments on both real and synthetic datasets.

38字×23行

38字×23行

38字×23行