How to Choose Combinations in a Join of Search Results

Mirit Shalem
Department of Computer Science, Technion
Haifa, Israel
mirit2s@cs.technion.ac.il

Yaron Kanza
Department of Computer Science, Technion
Haifa, Israel
kanza@cs.technion.ac.il

ABSTRACT

We present novel measures for estimating the effectiveness of duplication-removal operations over a join of ranked lists. We introduce a duplication-removal approach, namely *optimality rank*, that outperforms existing approaches, according to the new measures.

Categories and Subject Descriptors

H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval

General Terms

Algorithms, Experimentation, Measurement

Keywords

Data integration, search, top-k, join, diversity

1. INTRODUCTION

In complex search tasks it is often required to combine the answers of several basic search queries. Example 1.1 illustrates that.

EXAMPLE 1.1. *A businessperson plans a trip, under a restricted budget. She needs to find a hotel, and two restaurants near the hotel—one for lunch and one for supper. The sum of (1) the hotel's daily rate, and (2) the cost of a lunch and a supper should not exceed the daily budget. This complex search task comprises three basic search queries which return a ranked list of hotels, and two ranked lists of restaurants. These lists should be joined according to the budget and spatial constraints.*

Answering complex search tasks, such as those in the example above, is by *joining* the ranked lists that are the answers to the basic search queries—such join is similar to a relational join in databases. For the technical details on how to extract data from multiple sources, see [2]. The result of a join is a set of *combinations*. Each combination is a tuple of related items—comprising a single item from each list. In Example 1.1, the join produces triplets, of a hotel and two restaurants, that satisfy the geographic and the budget constraints. Combinations are ranked according to the *ranking scores* of the items they comprise.

Typically, in the result of a join there are many duplicate appearances of items. For instance, if in Example 1.1 some hotel has five restaurants suitable for lunch and three restaurants suitable for supper, it may appear fifteen times in the join result. Consequently, a join result can be too large to be useful—in fact, the number of combinations can be exponential in the size of the search answers.

Decreasing the size of the result is by discarding some of the combinations, i.e., by choosing a small subset of the result. We propose two measures for estimating the quality of result sets, and a novel semantics for choosing a subset. We compare experimentally our semantics to other common approaches, with respect to the proposed quality measures.

WWW 2011, March 28–April 1, 2011, Hyderabad, India.
ACM 978-1-4503-0637-9/11/03.

2. DEFINITIONS

Given a *combined search query* Q that comprises several basic subqueries, we denote by $\mathit{Join}(Q)$ the set of *all* the combinations that are produced by joining the answers to the subqueries of Q. The *score* of a combination is the result of a monotonic function, e.g. *sum*, over the item scores. Since the number of combinations can be large, we take a subset of it as the answer. Next, we formally define the measures that affect the selection of this subset.

Coverage. Coverage measures the percentage of "good" items that appear in the result. Let $(\ldots, o, \ldots)$ denote a combination that contains o. Given a set of combinations $\mathcal{C}$, we denote by $\mathit{Objects}(\mathcal{C})$ the set $\{o \mid \exists\ (\ldots, o, \ldots) \in \mathcal{C}\}$ of objects that appear in combinations of $\mathcal{C}$. Given a set $A \subseteq \mathit{Join}(Q)$ of combinations, the coverage of A is the ratio $\frac{|\mathit{Objects}(A)|}{|\mathit{Objects}(\mathit{Join}(Q))|}$.

Per-Item Optimality. In a join of search results, an item may appear in more than one combination. The *optimal* combination of an item o is the combination with the highest score, among the combinations that contain o. It is desired that every item will appear only in its optimal combination. There are two reasons to this: (1) this reduces duplications, which increases the effectiveness of the result by keeping its size small while maintaining diversity; and (2) having an item in a combination that is not optimal for it may mislead users. For instance, if in Example 1.1 a user sees a hotel in a non-optimal combination, she may consider this hotel as inappropriate, even when in its optimal combination it is suitable for her.

For measuring optimality, we define the *optimality count* of a combination C, denoted $\mathit{OptCount}(C)$, to be the number of items in C for which C is the optimal combination. Now, given a set $A \subseteq \mathit{Join}(Q)$ of n combinations of size t, the per-item optimality of A (PI-optimality, for short) is the ratio $(\sum_{C \in A} \mathit{OptCount}(C))/(nt)$, where $\sum_{C \in A} \mathit{OptCount}(C)$ is the number of appearances of an item in its optimal combination, in A, and nt is the total number of appearances of items in A (with repetitions).

One of the difficulties in answering complex queries is the conflict between maximizing coverage to maximizing PI-optimality.

3. SEMANTICS

We now present three common semantics and one novel semantics that specify which combinations to choose as the answer to a combined search query. We compare these semantics in terms of coverage and PI-optimality.

Top k. Choosing the k most highly ranked combinations is a simple and common approach. However, if the join produces many combinations, there are expected to be many repetitions of items in the result, and hence, both the coverage and the PI-optimality of the top-k result is expected to be low.

Skyline. The *skyline operator* [1] has been proposed as an alternative to top-k, where a *dominance relationship* is being defined to specify the combinations of the result. Let $C' = (o'^1, o'^2, \ldots, o'^t)$ and $C = (o^1, o^2, \ldots, o^t)$ be two combinations. We say that C' *dominates* C and write $C' \succ C$, if for all $1 \leq i \leq t$, $score(o'^i) \geq score(o^i)$, and exists $1 \leq j \leq t$ such that $score(o'^j) > score(o^j)$. The skyline operator returns only the combinations that are maximal with respect to dominance, i.e., given a query Q, $Skyline(Q) = \{C \mid C \in Join(Q)$ *and* $\nexists\, C' \in Join(Q)$ *such that* $C' \succ C\}$. For a relatively small number of lists, the skyline operator typically discards many combinations, and thus, the result has a low coverage.

RepeatedTop1 (diversifying). The *RepeatedTop1* semantics is commonly used to provide diversity in search results (see [3]). In RepeatedTop1, the answer is computed iteratively over a set of candidate combinations. Initially, $Join(Q)$ is the set of candidate combinations. In each iteration, the top-1 combination among the candidate combinations is added to the result, and any other combination that shares an item with the selected combination is discarded from the candidate set. When the candidate set is empty, the computation ends. RepeatedTop1 provides an answer which is diverse in the sense that each item appears at most once. However, in common scenarios, RepeatedTop1 has a low PI-optimality.

OptimalityRank. We propose a new semantics that, in addition to defining which combinations are selected, also defines their order. The combinations of $Join(Q)$ are sorted according to their optimality count (combinations with the same optimality count are sorted according to their scores). Essentially, only combinations with optimality count greater than zero are returned. Thus, every combination in the result is optimal for at least one of its items.

The number of combinations in the answer under this semantics is at least one (if the join is not empty) and it is at most the number of items in the ranked lists, e.g., a join of t ranked lists of size n produces at most nt combinations under this semantics. This guarantees that the size of the answer is not exponential in the size of the lists (*e.g.,* in $Join(Q)$, there can be n^t combinations). In addition, we prove, in the full version of this paper, that answers under the $OptimalityRank$ semantics are not necessarily contained in the result of the skyline operator, and not necessarily contain them.

$OptimalityRank$ provides full coverage, because, for any item o that appears in a combination of $Join(Q)$, the optimal combination of o is in $Join(Q)$. This semantics is also designed to provide high PI-optimality. The balance between coverage and PI-optimality can be tuned by limiting the size of the answer, *i.e.,* returning merely the top-k tuples, according to the optimality-count order. For a small k, PI-optimality is high, and as k increases, the PI-optimality decreases whereas the coverage increases.

4. EXPERIMENTAL EVALUATION

We evaluated and compared the aforementioned semantics in experiments over real data. Our data was obtained from Yahoo! Local using web services. We posed four queries and used a simple join condition which is based on the distance between the locations. We computed the result sets under the *Skyline*, *RepeatedTop1*, *OptimalityRank* semantics, and the full join, and we compared them in terms of size, coverage and PI-optimality. The sizes of *Join*, *OptimalityRank*, *Skyline*, and *RepeatedTop1* are 83196, 426, 7, and 45 combinations, respectively. Note that the result size of *OptimalityRank* is relatively high compared to the other sets, while the size of *Skyline* is the smallest. This is a major disadvantage of *Skyline* for combined search. Since the coverage increases when the result size increases, we conducted another evaluation, which compares sets of the same size. Given the result sets of *Skyline*, *RepeatedTop1*, *Join*, and *OptimalityRank*, we compare independently the coverage and PI-optimality of the *top-k combinations* in each set, as a function of k. Note that the top-k of the first three sets are determined simply by the combination scores, while *OptimalityRank* is sorted first by optimality count and only then by score. The results, which are shown in Figure 1 and Figure 2, show that *OptimalityRank* provides a better PI-optimality, for any value of k[1], than the other semantics. It also provides good coverage, although *RepeatedTop1* has the highest coverage as it forbids repetitions of items.

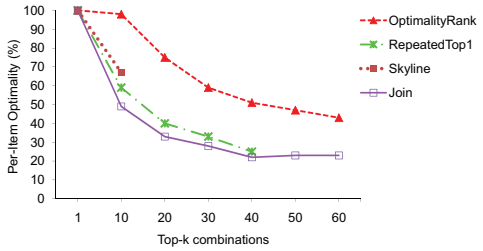

Figure 1: PI-optimality of result sets over Yahoo! Local data.

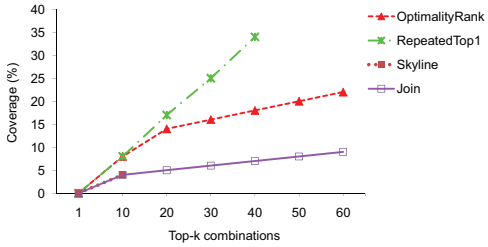

Figure 2: Coverage of result sets over Yahoo! Local data.

5. REFERENCES

[1] S. Borzsonyi, K. Stocker, and D. Kossmann. The skyline operator. *Data Engineering, International Conference on*, 0:0421, 2001.

[2] D. Braga, A. Campi, S. Ceri, and A. Raffio. Joining the results of heterogeneous search engines. *Information Systems*, 33(7-8):658–680, 2008.

[3] J. Carbonell and J. Goldstein. The use of mmr, diversity-based reranking for reordering documents and producing summaries. In *SIGIR*, pages 335–336, 1998.

[1]For *Skyline* and *RepeatedTop1* the corresponding lines are shown for values of k that are not greater than the size of the result.