Relaxation Paradigm in a Flexible Querying Context

Patrick Bosc, Allel HadjAli, and Olivier Pivert

IRISA/ENSSAT, Université de Rennes 1
6, rue de Kerampont – BP 80518, 22305 Lannion Cedex, France
{bosc, hadjali, pivert}@enssat.fr

Abstract. In this paper, we discuss an approach for relaxing a failing query in the context of flexible querying. The approach relies on the notion of proximity which is defined in a relative way. We show how such proximity allows for transforming a given predicate into an enlarged one. The resulting predicate is semantically not far from the original one and it is obtained by a simple fuzzy arithmetic operation. We show also how the search for a non-failing relaxed query over the lattice of relaxed queries can be improved by exploiting the notions of MFSs (*Minimal Failing Sub-queries*) and MGQs (*Maximally Generalized failing Queries*) of the original query.

1 Introduction

The rapid expansion of the Internet has made a variety of databases, including bibliographies, scientific databases, and travel reservation systems accessible to a large number of lay users. One of the common problems that users might be confronted with in their Web search is the *failing query problem*: users' queries return an empty set of answers. Users are frustrated by such kind of answers since they do not meet their expectations and interests. Their desire would be to find alternative answers that are related to the answers of the original queries. One technique that could enable for providing such answers is called *relaxation*. Query relaxation aims at expanding the scope of a query by relaxing the constraints involved in the query.

Let us note that manually relaxing failing queries is a tedious and time-consuming task. Since the late 80's, several automated approaches to query relaxation have been proposed [6]. The main goal of those approaches is to modify a failing user query into a relaxed query whose answer is non-empty, or at least to identify the cause of the failure. Some approaches are based on the concept of *false presupposition*. Recall that a *presupposition* of a statement is any statement entailed by the original. For instance, the statement "the king of France is bald" has as presupposition "there is a king of France" which is a *false* presupposition. Motro [10] has addressed the issue of empty answers by proposing a relaxation method which focuses on finding the false presuppositions of a failing query. Each generalization of the query is considered as a presupposition of the query. Query generalization is obtained by relaxing to a degree some of the conditions involved in the query. The system proposed can find all *Maximally Generalized failing Queries* (MGQs). Such responses have the potential to be too informative either to explain the failure or to help for turning the query into a non-failing one. A related approach has been proposed by Godfrey [7], who considers any sub-query (the query with some of its conditions eliminated) as a presupposition

H. Legind Larsen et al. (Eds.): FQAS 2006, LNAI 4027, pp. 39 – 50, 2006.

of the query itself. The focus of this work is the search for *Minimal Failing Sub-queries* (MFSs) of a failing query. Such sub-queries are the smallest sub-queries that fail. They constitute a better response to failing database queries.

In the context of flexible queries (i.e., queries that contain gradual predicates), the empty answer problem could still arise. Namely, there is no available data in the database that *somewhat satisfies* the user query. Only few works have been done for dealing with this problem. They mainly aim at relaxing the fuzzy requirements involved in the failing user query. This can be done by applying a transformation to some or all conditions of that query. A flexible query relaxation approach has been proposed by Bosc *et al.* in [2][3]. It is based on a particular *tolerance relation* modeled by a *relative proximity* parameterized by a tolerance indicator. This notion of proximity is intended for defining a set of predicates that are close, semantically speaking, to a given predicate P. In this paper, we show how flexible conjunctive queries can be relaxed locally, i.e. the relaxation is performed only on some sub-queries of the failing query, using a tolerance-based transformation. On the other hand, we also show how the notions of MGQs and MFSs could be beneficial for relaxing such queries. In particular, we propose an efficient search technique for finding a non-failing query by exploiting the MFSs of the original query.

The paper is structured as follows. Section 2 recalls some relaxation approaches that have been proposed in the Boolean context. In section 3, we describe the problem of flexible query relaxation on the one hand, and propose a method to solve this problem on the other hand. Section 4 shows how the MFSs can be used for finding a non-failing query over the lattice of relaxed queries in an efficient way. Last, we briefly recall the main features of our proposal and conclude.

2 Boolean Query Relaxation

Let us first introduce the following basic notions. A conjunctive Boolean query Q is of the form $Q = A_1 \wedge \ldots \wedge A_N$, where each A_i is an atomic condition. Q' is a sub-query of Q iff $Q' = A_{s_1} \wedge \ldots \wedge A_{s_m}$, and $\{s_1, \ldots, s_m\} \subset \{1, \ldots, N\}$. Let us recall that with a Boolean query, an item from the database simply either matches or it does not. Moreover, if a sub-query fails, then the query itself must fail.

As mentioned in the Introduction, we can distinguish two main approaches that can be used to relax a conjunctive Boolean query when it fails to produce any answer:

- Motro's approach: It consists in transforming query conditions into more general conditions;
- Godfrey's approach: It aims at generalizing a query by removing some parts from the query.

Both of the two approaches are based on the notion of *false presuppositions*.

2.1 Motro's Approach

In [10], Motro has proposed an approach to deal with failing queries. This approach combines the notion of relaxing queries into more general queries and that of searching for false presupposition. The query generalizations are obtained by replacing some query conditions by more general ones. In particular, generalization