Chapter 6

Conclusions

The objective of this research project was to propose and implement a solution to solve some challeges for data retrieving in ubiquitous computing environments. The solution is not related to some specific aplicative area (i.e. astronomy, bilogy, business, among others). It can be applied to meny different context if the validation results are successful.

6.1 Results and contributions

We propose a new query optimization technique that exploits case-based reasoning in order to improve query optimization in ubiquitous environments. Our approach deals with the challenge that the lack of metadata implies in this execution context. We propose a technique that is based on the useful knowledge (resource consumption measures) obtained from previous query executions. In addition, we propose a technique that allows the configuration of the optimization objective according to the users and application requirements, even for each single query.

The most important contributions of our work are the following:

- *The proposition of a data model for knowledge representation.* We propose a model to represent a query, a problem and a case. Moreover we define a similarity function to determine which query, related to cases stored in the case base, is similar and useful to optimize a new one (the problem to be solved).

- *Adaptation of case-based reasoning process to query optimization.* The solution is centered in the reasoning steps related to retrieval and re-adaptation of the useful knowledge. These steps retrieve the most relevant cases and adapt a previous solution to the new situation. The adaptation depends on the similarity level that is determined between the two queries. Basically, the adaptation process consists in evaluating the operations of the query with variations in their conditions.

- *Pseudo-random query plan generation technique.* For the generation of knew knowledge, we propose a pseudo-random query plan generaton technique able to propose solutions for new query problems. It is capable to construct new query plans and propose new solutions without using classical statistics and metadata to validate the query plans.

- *First prototype implementation.* We built SQuO, a query optimizer that implement our proposed solution. It is composed by two main modules, a case-based reasoner that involces four submodules that correspond to the four-step case based reasoner process approach. And, a query plan generator, its implementation is based on the pseudo-random query plan generation technique mentioned before.

The approach that we propose present some advantages that concerns directly with our objectives:

- Statistics are not needed. This optimization strategy can be applied where classical query optimization techniques can not. The optimization is progressive, the reason is that the learning is continuous, this means, as more we learn, a better optimization can be achieved.

- Optimization objective customization. This means that the optimization technique that we propose is not tied to optimize according just to execution time, as typically, classical query optimization techniques does, also means that the optimization objective can change for each query.

6.2 Perspectives and future work and

Currently we are working in a prototype that implements our solution in order to perform an experimental evaluation of our approach.

It is very interesting to apply this solution to different application domains where possible solutions can be enumerated, performance criteria are known, but where important information is not available when deciding the best solution (statistics in our case). For example, automatic service composition generated on demand via a declarative language seems a good target. Dynamicity management is also an important point to work on. However, the system should be able to detect those cases in its case base no longer relevant and thus delete them. This is a problem of knowledge refreshment. We are also working on improving the knowledge acquisition and exploitation, for example by sharing case bases between nodes in a network or by reducing the granularity of cases to handle sub queries instead of full queries.

As conclusion, future work involves:

- An implementation improvement and an extensive experimental evaluation of our prototype

- An improvements related to the knowledge acquisition and exploitation. For example: Divide the query and take the parts that can be useful to solve the new query Preload the case base with a set of queries already solved, and/or in a networks of nodes, where each node is a case base, share the knowledge contained in each of them.

- Dynamicity management. Related to reactions taken as a result of changes in the environment. And dynamicity related to knowledge management, refreshment.

- Finally, the transportation of this solution to different application domains. Where possible solutions can be enumerated, performance criteria are known but where needed information to decide what is the best solution are not available (statistics in our case). For example, automatic service composition generated on demand via a declarative language seems a good target.