

# Chapter 5

## CONCLUSIONS

This dissertation has addressed to three database management problems: distributed integrity constraint management, optimal materialized views, and optimal quasi-view decomposition. Those are interrelated, but their solution can be addressed individually. In the following, we present the contributions and conclusions, and some important future research areas.

Chapter 2 addresses the problem of deriving the best possible decompositions, both during design and at update time, more specifically, the contributions are as follows. First, we introduce a generic optimization framework for achieving best decompositions: the search space is the set of all *feasible safe* decompositions  $(C_1, \dots, C_M)$  of the global constraint  $\Omega$  over  $M$  distributed sites, i.e.,  $C_1 \wedge \dots \wedge C_M$  imply  $\Omega$ . The objective function can describe a variety of optimization criteria, such as the probability of an update satisfying local constraints, the expected number of updates before an update violates local constraints, or the average of the expected overall cost of manipulations during an update. Feasible decompositions are characterized by decompositions having the first

and possibly other properties from the following list): (1) safety, (2) local consistency, (3) partial constraint preservation, and (4) resource partition. One or more properties 1 through 4 are required for various decomposition scenarios, depending on what is known at the time of a decomposition.

Second, for the case of general linear arithmetic constraints, we reduce the optimization-based framework to a standard, finitely-specified problem of mathematical programming. To do that, we introduced the notion of *compact split* (safe) decompositions, and prove that for any monotonic objective function the optimal safe decomposition can always be found in the subspace of compact split decompositions. Then, we prove existence and actually developing a finite parametric (i.e., in terms of coefficients) characterization of the properties 1 through 4 of feasible decompositions together with optimization criteria.

Third, we develop an algorithmic framework to solve the resulting optimization problems. The constructed optimization problems, formulated in terms of parametric descriptions, have linear constraints and a non-linear objective function, based on a parametric representation of the volume function for the constraint space. For the design-time case, where each local constraint is in a single variable, the constructed objective function is concave; this property enables us to use a global search algorithm. We adopt the Frank-Wolfe algorithm to solve it. For other cases, the objective function is not concave and we use local search techniques in the algorithmic framework, that incorporate the Frank-Wolfe algorithm for search in local neighborhoods. To run

experiments and to show the feasibility of the approach, we have implemented an optimization engine for the schema-based full decompositions with local constraints in single variables. The experiments suggest that the approach is feasible and scalable, but more experimental study will be necessary to fine-tune the algorithms for specific cases.

Chapter 2 also proposes a general framework to manage global linear arithmetic constraints in distributed databases, extending [BGM92]. This framework can be applied to different network and system architectures (centralized, hierarchical, and fully distributed). The framework considers two types of sites: coordinators and non-coordinators. A coordinator is a site that coordinates the constraint decomposition to facilitate a local update at a non-coordinator site  $k$  that violates the current local constraint. This is done by finding a set of sites  $\theta$ , containing site  $k$ , and trying to create a new (partial) compact split that would satisfy the new database state (i.e., new update). We propose a primitive RESOURCE-TRANSFER (a distributed transaction involving two sites) to deal with inter-site communication. We proved that under standard transaction management properties, any protocol that uses exclusively this primitive guarantees local and global consistency (i.e., current  $C_1, \dots, C_M$ , and  $\Omega$ ), it is resilient to failures, and produces an optimal decomposition for sites in  $\theta$ . Finally, we exemplify an instance of our distributed framework for the single coordinator case.

Chapter 3 introduces a generic optimization framework to decide optimally materialized views. First, we extend the expression-DAG (Direct Acyclic Graph) [RSS96] as

a mechanism to represent equivalent view evaluation plans. We characterize paths and the transitive closure of an expression-DAG (i.e., an expression-DAG with all possible equivalent view evaluation plans). We show that, under certain conditions, expression-DAG and AND-OR graphs are equivalent. However, while the size of a standard AND-OR graph is defined in terms of its nodes and arcs, the size of an expression-DAG is defined in terms of the cardinality of its operational nodes. Second, the optimization framework is formulated in terms of the expression-DAG structure. Thus, under certain objective function conditions, the problem of optimal selection of materialized views can be formulated as the constrained shortest path in an expression-DAG, i.e., a complete expression-path or AND-path. For this case, a linear-time algorithm (in terms of the expression-DAG size) is presented. Note that this special case can handle important applications such as inventory control and logistics support. We have performed some experiments, and the results suggest that our approach is feasible. Third, for the general optimization problem and if the expression-DAG has all possible view evaluation plans, then the linear-time algorithm can be applied to obtain a solution. Note that this algorithm does not evaluate all possible equivalent evaluation plans, because all those which are subsumed by others are eliminated earlier. Finally, if the expression-DAG with all possible view evaluation plans is not available, a local search algorithm is presented. However, further research is necessary in this area.

In Chapter 4, we formalize and extend considerably the notion of quasi-view (a view with explicit re-materialization conditions, called *refresh conditions*) to multi-databases

and create an optimization framework to design them. First, we show that the optimal quasi-view decomposition problem is not a separable problem, i.e., it has to be considered as both view decomposition and refresh condition evaluation together. Second, a general solution strategy is proposed, which introduces the notion of a conditional problem (i.e., optimization problems where some of its variables are fixed), where the optimization problem is reduced to the optimal view materialization and the constraint decomposition problems. Third, for the special case of disjunctive refresh conditions, we prove that the conditional refresh conditions decomposition problem is equivalent to finding a compact split (safe) decomposition, and therefore, all results from Chapter 2 can be applied.

Finally, this dissertation has presented an integrated framework for the analysis, design, and optimal decomposition of global database constraints, views, and quasi-views. This approach is a contribution to the state-of-the-art in constraint, view, and quasi-view management, in that the general decomposition problems can be formulated as optimization problems.

## **Future Work**

This dissertation provides a framework to decompose optimally constraints, views and quasi-views, and it has been shown that this approach is formal, rigorous and feasible. However, there are many possible extensions to this work.

In the area of constraint decomposition, the most natural is to extend the results

to more generic numeric constraints, and to provide effective optimization algorithms. This is an important topic, since arithmetic linear constraints might be very restrictive for some domains. A second direction is to explore different objective functions, in which other criteria may be used. A third direction is to consider replicated data, i.e., relaxing the assumption that the variables in the constraint must belong to a partition; this is of special interest in practical problems.

The framework to manage constraints has based its results for local transactions (especially local updates). However, when a distributed transaction is processed the inter-site coordination problem must be solved. Another area is to implement effectively other architecture configurations, especially distributed ones. These instantiations could be used to solve problems related to multiple resource requirements.

In terms of materialized views (Chapter 3), the local search algorithm needs to be explicitly implemented and studied in terms of its complexity. A very important issue is related to the transitive closure of an expression-DAG, and whether it can be derived efficiently from an expression-DAG, the constrained shortest path problem represents a good alternative. Finally, research is needed to identify the classes of problems to which our approach can be applied efficiently, for example conjunctive views.

Finally, our quasi-view work can be extended directly to provide a more sophisticated semantics, i.e., extending the action part of the ECA paradigm from simply refreshing a quasi-view, to executing a set of rules. Also, this work can be extended to different types of refresh conditions, for instance, conjunctive ones.