

# Web Service Composition Through Declarative Queries: The Case of Conjunctive Queries with Union and Negation\*

Bertram Ludäscher<sup>†</sup>

Alan Nash<sup>‡</sup>

## 1. Web Service Relations with Access Patterns

A *web service operation* can be seen as a function  $op: X_1, \dots, X_n \rightarrow Y_1, \dots, Y_m$  having an *input message (request)* with  $n$  arguments (*parts*), and an *output message (response)* with  $m$  parts [4, Section 2.2]. For example,  $op_B: \text{Author} \rightarrow \{(\text{ISBN}, \text{Title})\}$  may implement a books search service, returning for a given author  $A$  a list of books authored by  $A$ . We propose to model such operations as *relations with access pattern*, here:  $B^{\text{io}}(\text{ISBN}, \text{Author}, \text{Title})$ , where ‘io’ indicates that a value for the second attribute must be given as *input*, while the other attribute values can be retrieved as *output*. In this way, a family of web service operations over  $k$  attributes can be concisely described as a relation  $R(A_1, \dots, A_k)$  with an associated set of access patterns  $\mathcal{P} \subseteq \{i, o\}^k$ . Since we model web services as relations with access patterns, we can now use *queries as a declarative specifications for web service composition*: e.g., the query  $Q(I, A, T) \leftarrow B(I, A, T), C(I, A), \neg L(I)$  asks for books available through a store  $B$  which are contained in a catalog  $C$ , but not in the local library  $L$ . Let the only access patterns be  $B^{\text{ioo}}$ ,  $B^{\text{io}}$ ,  $C^{\text{oo}}$ , and  $L^o$ . If we try to execute  $Q$  from left to right, neither pattern for  $B$  works since we either lack an ISBN  $I$  or an author  $A$ . However,  $Q$  is *feasible* since we can execute it by first calling  $C(I, A)$  which binds both  $I$  and  $A$ ; after that, calling  $B^{\text{ioo}}$  (or  $B^{\text{io}}$ ) will work. Calling  $\neg L(I)$  first and then  $B$  does not work: a negated call can only filter out answers, but cannot produce any new variable bindings.

## 2. Formal Results

We study the problem of deciding whether a query  $Q$  is *feasible*, i.e., whether there exists a logically equivalent query  $Q'$  that can be executed observing the limited access patterns given by the web service (source) relations. Executability depends on the specific syntactic form of a query, while feasibility is a more “robust” semantic notion, in-

volving all equivalent queries (i.e., reorderings, minimized queries, etc). Li shows that deciding query feasibility (called “stability” in [1]) is NP-complete for conjunctive queries (CQ) and for conjunctive queries with union (UCQ) [1]. An algorithm is presented which may, for certain instances, compute complete answers to queries that are not feasible. In [3] we extend these results to conjunctive queries with negation (CQ<sup>¬</sup>) and unions of conjunctive queries with negation (UCQ<sup>¬</sup>) and show that for both classes deciding feasibility is  $\Pi_2^P$ -complete. We also achieve a uniform treatment of CQ, CQ<sup>¬</sup>, UCQ, and UCQ<sup>¬</sup> by the notion of *answerable part*  $\text{ans}(Q)$  of  $Q$ , which for those classes is shown to be the minimal feasible query containing  $Q$ :

**Theorem 1** *If  $Q, E \in \text{UCQ}^\neg$  satisfy  $Q \sqsubseteq E$  and  $E$  is executable then  $Q \sqsubseteq \text{ans}(Q) \sqsubseteq E$ . That is,  $\text{ans}(Q)$  is a minimal executable (and thus also a minimal feasible) query containing  $Q$ .*

Feasibility is thus closely related to *query containment* ( $Q \sqsubseteq Q'$ ); indeed deciding feasibility is as hard as deciding containment for a large number of first-order fragments including universal queries [2].

We also show how to avoid the worst-case complexity for UCQ<sup>¬</sup>, both by approximations at compile-time and by a novel runtime processing strategy [3]. The costly containment check may sometimes be avoided by using two efficiently computable approximate execution plans  $Q^u$  and  $Q^o$ , which produce underestimates and overestimates of the actual answer for  $Q$ . A novel runtime algorithm may report complete answers even in the case of infeasible plans, and can sometimes quantify the degree of completeness [3], extending a similar technique for CQ [1].

## References

- [1] C. Li. Computing complete answers to queries in the presence of limited access patterns. *Journal of VLDB*, 12:211–227, 2003.
- [2] A. Nash and B. Ludäscher. Processing first-order queries with limited access patterns. *submitted for publication*, 2004.
- [3] A. Nash and B. Ludäscher. Processing unions of conjunctive queries with negation under limited access patterns. In *Intl. Conf. on Extending Database Technology (EDBT)*, 2004.
- [4] Web services description language (WSDL) Version 1.2. <http://www.w3.org/TR/wsdl12>, June 2003.

\* Work supported by NSF-ACI 9619020, NSF-ITR 0225676, NSF-ITR 0225673, and DOE/SciDAC DE-FC02-01ER25486. Authors' contact: <sup>†</sup>ludaesch@sdsc.edu, San Diego Supercomputer Center, UCSD and <sup>‡</sup>anash@math.ucsd.edu, Dept. of Mathematics, UCSD