Declarative Data-Driven Coordination

Johannes Gehrke
Department of Computer Science, Cornell University

With Gabriel Bender, Nitin Gupta, Lucja Kot, Sudip Roy (Cornell) and Milos Nikolic, Christoph Koch (EPFL)
• Assume Tom and Meg want to coordinate itineraries
  - Fly on the same flight, in adjacent seats
  - Also stay in the same hotel if possible
Coordination: Enrollment

Students want to enroll in classes with their friends

• Help with homework/moral support
• Already happens with out-of-band communication
• CourseRank
  - “Connect to Facebook to find out who of your friends is enrolled”

Interesting coordination scenarios:

• Negative constraints
  - Avoid the section my ex-* is in
• Strong mutual dependencies
  - I will take this tough class only if my friend takes it too
Coordination: MMOs

- Players want to form alliances with others based on shared or complementary goals
  - I will attack from the North if someone else attacks from the South

- These alliances may be completely ad-hoc and formed with total strangers just for the purpose of achieving one goal
Room Sharing among attendees of the 2011 ACM SIGMOD Conference

The conference officers have set up a web page where interested attendees of the conference can register their interest in sharing rooms at the conference hotel. Through this service attendees can enter their details so that interested people can contact each other.

To register your interest, please submit your information at: [http://bit.ly/sigm_share_room](http://bit.ly/sigm_share_room) (URL shortener service forwarding to a Google Spreadsheets form). This service is provided solely as a convenience to participants that seek to share accommodation costs. Please contact directly participants that have expressed interest. The organizers will not be involved in the process nor are they responsible for possible abuse of the information you provide.
Sharing a room at the Conference Hotel?

This form allows people who want to stay at the conference hotel to express their interest in sharing rooms. Please fill out the following form, all people expressing interest in sharing a room can then contact each other by looking at the following page: http://bit.ly/sigmod_share_room_list

* Required

Name *

Email *

Period you wish to stay at the hotel *

Please add any constraints on sharing a room (gender, etc)

Submit

Powered by Google Form
### Sharing a room at the Conference Hotel? : List

<table>
<thead>
<tr>
<th>Timestamp</th>
<th>Name</th>
<th>email</th>
<th>Period you wish to stay at the hotel</th>
<th>Please add any constraints on sharing a room (gender, etc)</th>
</tr>
</thead>
<tbody>
<tr>
<td>5/10/2011 6:15:25</td>
<td>Penny Xu</td>
<td><a href="mailto:penny@xyz.com">penny@xyz.com</a></td>
<td>12/06/2011-16/06/2011</td>
<td>Female</td>
</tr>
<tr>
<td>5/10/2011 6:38:39</td>
<td>Tom</td>
<td><a href="mailto:tom@abc.com">tom@abc.com</a></td>
<td>June 13 - June 17 (4 nights)</td>
<td>thanks!</td>
</tr>
<tr>
<td>5/10/2011 16:38:58</td>
<td>Kevin Johnson</td>
<td><a href="mailto:kevin@def.com">kevin@def.com</a></td>
<td>12-17 June</td>
<td>Males only. I already have a room reservation -- looking to fill the other bed and split the cost.</td>
</tr>
<tr>
<td>5/10/2011 18:34:49</td>
<td>Jane</td>
<td><a href="mailto:jane@ghi.com">jane@ghi.com</a></td>
<td>5 nights June 12-16 (inclusive)</td>
<td>prefer females (I'm a girl)</td>
</tr>
<tr>
<td>5/12/2011 13:03:30</td>
<td>Tyler</td>
<td><a href="mailto:tyler@jkl.com">tyler@jkl.com</a></td>
<td>13th-16th June</td>
<td>n/a</td>
</tr>
<tr>
<td>5/13/2011 12:45:54</td>
<td>Sarah</td>
<td><a href="mailto:sarah@lmn.com">sarah@lmn.com</a></td>
<td>12-17 June</td>
<td>I'm easy going :)</td>
</tr>
<tr>
<td>5/14/2011 12:20:15</td>
<td>Emily</td>
<td><a href="mailto:emily@opq.com">emily@opq.com</a></td>
<td>Check-in 11, Check-out 15</td>
<td></td>
</tr>
<tr>
<td>5/23/2011 22:47:20</td>
<td>George</td>
<td><a href="mailto:george@rst.com">george@rst.com</a></td>
<td>12th-17th</td>
<td></td>
</tr>
<tr>
<td>5/25/2011 23:16:36</td>
<td>Andrew</td>
<td><a href="mailto:andrew@tuv.com">andrew@tuv.com</a></td>
<td>June 12 - June 17</td>
<td>male</td>
</tr>
<tr>
<td>6/4/2011 13:10:08</td>
<td>Michael</td>
<td><a href="mailto:michael@wxyz.com">michael@wxyz.com</a></td>
<td>June 12th</td>
<td>Gender: male</td>
</tr>
</tbody>
</table>

Published by [Google Docs](https://docs.google.com) – [Report Abuse](https://www.google.com) – Updated automatically every 5 minutes
Coordination: Other Examples

• Scheduling meeting times with students/advisees
• Wedding gift purchases
  - People can group together to purchase a more expensive item
• Post-disaster emergency management
• Charity fund-raising with matching funds
• Joint gift-giving
It is not just the applications that are data-driven....

The coordination itself is data-driven too!
- Users want to agree on a choice of data values
- Not on the time of day of when jointly enrolling in a course

Today typically achieved with out-of-band communication
- Or through an ad-hoc solution for a given scenario...
D3C: Declarative Data-Driven Coordination

- Goal: Provide a declarative abstraction and mechanism to support D3C
  - Being declarative is fundamental principle in query and update languages
  - Coordination pertains to data, so should be expressed at the same level
  - Meg says: “Book me a ticket on the same flight as Tom”
  - System takes care of the actual coordination
D3C and The Legacy of Transactions

ACID Properties of a transaction

- Atomicity
- Consistency
- Isolation
- Durability

D3C requires relaxing isolation

- For semantic reasons, not for performance (such as lower isolation levels, eventual consistency)
- We still want atomicity and durability

And the communication due to coordination should be “controlled“

- “Residual" isolation
Existing Abstractions

• Operating systems:
  - Message passing
  - Shared memory
  - Transactional memory

• Programming languages:
  - Powerful formalisms such as the $\Pi$-calculus (channels)
  - Concurrent ML
  - Concurrent Haskell
  - Erlang

• Multi-agent systems
Existing Abstractions (Contd.)

Sagas/nested transactions

- Transactions inside transactions; only commit when outermost transaction commits

Triggers

- Program automatically executed as a response to certain events in the database
- Used instead of or after an INSERT, UPDATE, or DELETE operations on database tables

CREATE OR REPLACE TRIGGER trigger_name
BEFORE DELETE OR INSERT OR UPDATE ON table_name
FOR EACH ROW ...
Why A New Abstraction?

• Need an abstraction that is at the “right” level
  - Data-centric, not process-centric
  - Should not require users to manipulate low-level constructs like channels

• Hide the implementation of the coordination logic
  - The matching that must take place between different users’ coordination constraints

• We want to enable complex kinds of coordination
  - Mutual handshake: I will take this tough course but only if my friend does, and vice versa
Outline

- Introduction
- **Entangled queries**
  - Language
  - Query Evaluation
- Entangled transactions
- Open Problems
Outline

• Introduction
• Entangled queries
  - Language
  - Query Evaluation
• Entangled transactions
• Open Problems
Entangled Queries

Entangles queries: an abstraction and a mechanism for D3C

Example scenario: Steve and Larry want to travel to NYC on the same flight

- In addition, Steve wants to travel only on United
SELECT ‘Steve’, fno INTO ANSWER Reservation
WHERE
  fno IN (SELECT fno FROM Flights WHERE dest=‘JFK’)
  AND (‘Larry’, fno) IN ANSWER Reservation
CHOOSE 1
Steve’s Query

SELECT ‘Steve’, fno INTO ANSWER Reservation
WHERE
  fno IN (SELECT fno FROM Flights WHERE dest=‘JFK’)
AND (‘Larry’, fno) IN ANSWER Reservation
CHOOSE 1

• Larry’s answer must also be in the Reservation table
Larry’s Query

SELECT ‘Larry’, fno INTO ANSWER Reservation
WHERE
    fno IN (SELECT fno FROM Flights F, Airlines A
           WHERE F.dest='JFK' and F.fno = A.fno AND
           A.airline = ‘United’ )
    AND (‘Steve’, fno) IN ANSWER Reservation

CHOOSE 1

SELECT ‘Steve’, fno INTO ANSWER Reservation
WHERE
    fno IN (SELECT fno FROM Flights WHERE dest='JFK')
    AND (‘Larry’, fno) IN ANSWER Reservation

CHOOSE 1
## Flights Database

<table>
<thead>
<tr>
<th>Flight</th>
<th>Destination</th>
</tr>
</thead>
<tbody>
<tr>
<td>122</td>
<td>JFK</td>
</tr>
<tr>
<td>123</td>
<td>JFK</td>
</tr>
<tr>
<td>134</td>
<td>JFK</td>
</tr>
<tr>
<td>136</td>
<td>Brussels</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Airlines</th>
<th>Flightno</th>
<th>Airline</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>122</td>
<td>United</td>
</tr>
<tr>
<td></td>
<td>123</td>
<td>United</td>
</tr>
<tr>
<td></td>
<td>134</td>
<td>Lufthansa</td>
</tr>
<tr>
<td></td>
<td>136</td>
<td>Alitalia</td>
</tr>
</tbody>
</table>
United Flights 122 and 123 satisfy the constraints.
Query Evaluation

• Individual queries do not see the full ANSWER table, but are guaranteed that their constraints are satisfied.

• Both transactions that contain these entangled queries can now proceed and make bookings.

• Note that the coordination partner was specified implicitly using data values, not explicitly.
SELECT select_expr
INTO ANSWER tbl_name [, ANSWER tbl_name] ...
FROM TABLE
[WHERE answer_condition]
CHOOSE 1

- Currently, we allow only SPJ (conjunctive) queries in the WHERE clause
  - Could be extended with disjunction, union, aggregate constraints, ...
Outline

• Introduction
• Entangled queries
  - Language
  - Query Evaluation
• Entangled transactions
• Open Problems
Evaluating Entangled Queries

How do we evaluate entangled queries?

Problem: Evaluation is NP-complete in the general case
• Not that surprising: Entangled queries can encode CSP

More than one source of complexity:
• Matching up the entangled queries
• Finding data values that satisfy coordination constraints
Stages of query evaluation

1. Check queries for safety
2. Partition queries into subsets and match queries
3. Create and evaluate the combined query and construct the individual answers
Stages of query evaluation

1. Check queries for safety
2. Partition queries into subsets and match queries
3. Create and evaluate the combined query and construct the individual answers
In many settings, there will be only one way to match up the queries for coordination

- Specify this formally as a notion of safety for a set of queries
- Test for safety
A Datalog-like representation (without recursion)

\{ C \} \quad H :\!-\! B

C, H and B are conjunctions of relational atoms
- C and H over answer relations
- B over database (non-answer) relations

Representation of Larry and Steve's queries
- \{ Booking(Larry,x) \}  Booking(Steve,x) :\!-\! Flight(x, JFK)
- \{ Booking(Steve,y) \}  Booking(Larry,y) :\!-\! Flight(y, JFK)
  \Lambda\ \text{Airline}(y, \text{United})
Safety

A set of queries is unsafe if there a query with more than one potential coordination partner

\{
\text{Booking (Larry, x)}\}

\text{Booking (Steve, x) :- Flight (x, JFK)}

\{
\text{Booking (Larry, x)}\}

\text{Booking (Bill, x) :- Flight (x, JFK)}

\{
\text{Booking (u, x)}\}

\text{Booking (Larry, x) :- Friend (Larry, u), Flight (x, JFK)}

Safety is independent of data
- Asking the system to choose between users is different from asking it to choose between flight numbers
- Safety is formalized using logical unifiability between heads and postconditions
Two relation atoms (referring to the same relation) are unifiable unless they contain different constants in the same attribute
- \( R(x; y) \) and \( R(z; z) \) are unifiable
- \( R(2; y) \) and \( R(3; z) \) are not

Query \( q \) is a potential coordination partner for \( q' \) if some head atom of \( q \) unifies with some postcondition atom of \( q' \).

A set of queries is **unsafe** if there a query with more than one potential coordination partner

Simple algorithm: Iterate over query set and search for queries with postconditions that unify with heads from more than one query.
Query Evaluation

Stages of query evaluation

1. Check queries for safety
2. Partition queries into subsets and match queries
3. Create and evaluate the combined query and construct the individual answers
Partitioning and query matching use a structure called the unifiability graph:
- One node per query
- Edge from q to q’ if a head atom of q unifies with a postcondition atom of q’

Example:
- q1 : \{R(x1) \land S(x2)\} \quad T(x3) \quad :- \quad D1(x1; x2; x3)
- q2 : \{T(1)\} \quad R(y1) \quad :- \quad D2(y1)
- q3 : \{T(z1)\} \quad S(z2) \quad :- \quad D3(z1, z2)
Matching

Unifiability graph gives overall structure of how queries match up

But we know more information:

- $q_1 : \{R(x1) \land S(x2)\}$  
  $T(x3) \quad :- \quad D1(x1; x2; x3)$
- $q_2 : \{T(1)\}$  
  $R(y1) \quad :- \quad D2(y1)$
- $q_3 : \{T(z1)\}$  
  $S(z2) \quad :- \quad D3(z1, z2)$

- The head of $q_1$ only satisfies the postcondition of $q_2$ if $x3=1$
  - Eventually, all the variables will be associated with values from the DB, so we will have a valuation
  - We know coordination is only possible for valuations that assign $x3$ the value 1
• Represent this information as unifiers associated with nodes in the graph
• A unifier is a constraint imposed by a particular query
  - q1 : \{R(x_1) \land S(x_2)\}  \quad T(x_3) :- D_1(x_1, x_2, x_3)
  - q2 : \{T(1)\}  \quad R(y_1) :- D_2(y_1)
  - q3 : \{T(z_1)\}  \quad S(z_2) :- D_3(z_1, z_2)
Matching (Contd.)

- Suppose a head of q unifies with a postcondition of q’
  - q’ “relies” on q for satisfaction
  - q is unique for this q’, by safety
  - so, if q’ is to receive an answer, q must receive an answer too
  - so, any valuation constraints from q apply to q’ as well!
Matching (Contd.)

Query matching is an iterative process that propagates these unifiers through the graph

- Related to the chase and to arc-consistency
- May remove nodes from the graph (queries whose postconditions cannot be satisfied)
- Eventually either fails or reaches a fixpoint \( \rightarrow \) matching

\[
\{x_1, y_1\}, \{x_2, z_2\}, \{z_1, x_3, 1\}
\]

\[
\{x_1, y_1\}, \{x_2, z_2\}, \{z_1, x_3, 1\}
\]
Stages of query evaluation

1. Check queries for safety
2. Partition queries into subsets and match queries
3. Create and evaluate the combined query and construct the individual answers
Building the Combined Query

\{
\text{Booking}(\text{Larry}, \, x)\}\ \text{Booking}(\text{Steve}, \, x) \, :\, - \, \text{Flight}(x, \, \text{JFK})$

\{
\text{Booking}(\text{Steve}, \, y)\}\ \text{Booking}(\text{Larry}, \, y) \, :\, - \, \text{Flight}(y, \, \text{JFK})$

\quad \land \, \text{Airline}(y, \, \text{United})$

\text{Gets rewritten to:}$

\text{Booking}(\text{Larry}, \, x) \, \land \, \text{Booking}(\text{Steve}, \, x) \, :\,$

\quad \text{Flight}(x, \, \text{JFK}) \, \land \, \text{Airline}(x, \, \text{United})$
Experimental Setup

- Prototype implemented in Java and uses JDBC to connect to a MySQL database system

Dataset:
- Generate queries that match in pairs or triples
- Make queries more or less specific (coordinate with a named friend vs. any friend)

Additional experiments:
- Increase number of post-conditions per query
- Stress-test performance of matching algorithm
Results: Scalability

![Graph showing scalability results for 2-set, generic, 2-set, specific, and 3-set, specific scenarios.](image)
Increasing # of constraints

![Graph showing processing and query times with increasing number of post-conditions.](image)
Results: Concurrency
Outline

• Introduction
• Entangled queries
• Entangled transactions
• Open Problems
Entangled Transactions

- **Goal:** Extend transactions to incorporate entangled queries

- **Challenge:**
  - Relationship of entanglement to classical transactions

- **Example scenario**
  - Steve and Larry want to fly together to NYC
  - If they can make a flight booking together, then they want to stay in the same hotel
BEGIN TRANSACTION WITH TIMEOUT 2 DAYS;

SELECT `Steve`, fno, fdate AS @ArrivalDay INTO ANSWER FlightReservation
WHERE fno, date IN (SELECT fno, fdate FROM Flights WHERE dest=`JFK')
AND (`Larry', fno, fdate) IN ANSWER FlightReservation
CHOOSE 1;

-- (Code to perform flight booking omitted)

SET @StayLength = `2011-10-30' - @ArrivalDay;

SELECT `Steve', hid, @ArrivalDay, @StayLength INTO ANSWER HotelReservation
WHERE hid IN (SELECT hid FROM Hotels WHERE location=`NYC')
AND (`Larry', hid, @ArrivalDay, @StayLength) IN ANSWER HotelReservation
CHOOSE 1;

-- (Code to perform hotel booking omitted)

COMMIT;
Recall consistency:
• Every transaction, if executed by itself on an initially consistent database, will produce another consistent database.

What is the analogous property for entangled transactions --- what is a unit of work?
• Portions of a transaction?
• A transaction?
• A group of transactions?

BEGIN TRANSACTION WITH TIMEOUT 2 DAYS;

SELECT `Steve`, fno, fdate AS @ArrivalDay INTO ANSWER FlightReservation
WHERE fno, date IN (SELECT fno, fdate FROM Flights
WHERE dest=`JFK')
AND (`Larry', fno, fdate) IN ANSWER FlightReservation
CHOOSE 1;

-- (Code to perform flight booking omitted)

SET @StayLength = `2011-10-30' - @ArrivalDay;

SELECT `Steve', hid, @ArrivalDay, @StayLength INTO ANSWER HotelReservation
WHERE hid IN (SELECT hid FROM Hotels WHERE location=`NYC')
AND (`Larry', hid, @ArrivalDay, @StayLength) IN ANSWER HotelReservation
CHOOSE 1;

-- (Code to perform hotel booking omitted)

COMMIT;
Consistency

**Entangled Query Oracle**

- Process that executes alongside an entangled transaction
- For an entangled query, the oracle chooses a valid answer (=ground the query on the database) and returns it to any entangled query
- Has no direct effect on the database’s state

**Oracle Consistency:**

- Suppose an entangled transaction executes by itself on an initially consistent database, **using an entangled query oracle to obtain answers to the entangled queries**. Then the execution will produce another consistent database.
Isolation

Anomaly-based definition

Two anomalies

• Widowed Transactions
• Unrepeatable quasi-reads

Let us review first what we mean by anomalies for classical transactions.
• Consider a possible interleaving (schedule) of two transactions:

| T1: | A=A+100, B=B-100 |
| T2: | A=1.06*A, B=1.06*B |

• The system’s view of the schedule:

| T1: | R(A), W(A), R(B), W(B) |
| T2: | R(A), W(A), R(B), W(B) |
Scheduling Transactions

• Serial schedule: Schedule that does not interleave the actions of different transactions.
• Equivalent schedules: For any database state
  - The effect (on the set of objects in the database) of executing the schedules is the same
  - The values read by transactions is the same in the schedules
• Serializable schedule: A schedule that is equivalent to some serial execution of the transactions.

• Note: If each transaction preserves consistency, every serializable schedule preserves consistency.
• Reading Uncommitted Data (WR Conflicts, “dirty reads”):

T1: \( R(A), W(A), \) \hspace{1cm} R(B), W(B), Abort
T2: \hspace{1cm} R(A), W(A), C

• Unrepeatable Reads (RW Conflicts):

T1: \hspace{1cm} R(A), \hspace{1cm} R(A), W(A), C
T2: \hspace{1cm} R(A), W(A), C
Overwriting Uncommitted Data (WW Conflicts):

T1: \( W(A), W(B), C \)

T2: \( W(A), W(B), C \)
Isolation for D3C

Two new anomalies

• Widowed Transactions
• Unrepeatable quasi-reads
New Anomaly 1: Widowed Transactions

- Flight entangled queries
- Ticket booking code
- Hotel entangled queries
- Room booking code

ABORT
New Anomaly 2: Unrepeatabile Quasi-Reads

Entangled queries

Read Flights

Read Flights, Airlines

Entangled query evaluation

SELECT * FROM Airlines
WHERE airline = `United`

INSERT INTO Airlines
VALUES (125, `United`)
Eliminating These Anomalies

• How to avoid widowed transactions?
  - *Group commit* of all the transactions that are connected through entangled queries

• Unrepeatable quasi-reads
  - *Appropriate locking* of data structures
Scheduling

Steve
System evaluates all three flight queries

Books flight

Larry

Bill

System evaluates Mike and Stan’s hotel queries and Opher’s flight query

Books hotel

COMMIT

Steve’s transaction ready to commit, waits for Larry

COMMIT

ABORT

Bill’s transaction blocked, waits for retry of flight query
Putting Everything Together

- **Traditional ACID Properties:**
  - Atomicity
  - Consistency $\rightarrow$ Oracle Consistency
  - Isolation $\rightarrow$ Two new phenomena
  - Durability

- **We can now define**
  - Oracle-serializability: Serial schedule with a suitable oracle that provides answers to entangled queries
  - Entangled isolation: Schedule does not have any anomalies

- **Main theorem:** Any schedule that is entangled-isolated is also oracle-serializable.
Outline

- Introduction
- Entangled queries
- Entangled transactions
- Open Problems
Many exciting research directions:

• Extending the language for entangled queries
• Studying the complexity of evaluation
• Modeling entangled transactions
• Designing a system for end-to-end support of entangled queries
• Privacy and security issues
• Wider implications for system design of relaxing isolation
• Entanglement for CEP
SELECT P.partyid, 'Simon' INTO ANSWER Attendance
FROM Parties P
WHERE P.pdate='Friday'
    AND
(SELECT COUNT(*)
    FROM ANSWER Attendance A, Friend F
    WHERE P.partyid = A.partyid AND A.name = F.name1
    AND F.name2 = 'Paul') > 2

CHOOSE 1
Language Extensions (Contd.)

- "Soft" constraints
  - Travel dates should be as close as possible (but need not be identical)

- Preferences
  - Will travel on any US carrier, but prefer United if possible
  - Will travel any day next week, but the earlier the better

- Semantics where more than one record is returned
  - Example: Course enrollment
Complexity

- We need to understand the complexity of evaluation better
  - How do the different sources of complexity interact?

- What answering guarantees can we provide and when?
  - Do we always find an answer if one exists?
  - Do we find an answer that involves a maximal number of queries?

- How do language extensions affect tractability?
Supporting Entanglement in a System

Where does evaluation take place?
• Inside or outside the DBMS?

How to reconcile asynchronous query submission, synchronous query answering?
• Staleness
• Incremental evaluation strategies for coordination
System Support for Entanglement

Travel Application

- Visual Interface
- Application Logic
  - Handle Travel Booking Functionality
- Coordination

Command-line SQL interface

Facebook

Friend List

Notifications

Youtopia

- Query Compiler
- Execution Engine
- Coordination Component
- • Data
- • Coordination State

Select Coordination Strategy

Display State

Presentation Tier

Middle Tier

Coordination Requests (IR)

Coordination Requests (Extended SQL)
Privacy and Security

How much of the coordination information should be visible to whom?

- Should the whole answer relation be visible to everyone?
- Should we make the waiting queries visible to other users in order to enable them to join?

Is it possible to perform malicious attacks involving coordination?

- Make someone coordinate with an unintended partner
- Flood the system with queries to prevent coordination
Final Thoughts

What is the impact of relaxing isolation on the design of a data-driven systems?

- Isolation has been a cornerstone of the transaction abstraction for a very long time
- It permeates all aspects of database design
- Until today, if isolation has been relaxed, it has been for performance reasons rather than semantic reasons
Summary

• Many applications require some form of coordination between users
• This coordination should happen at the same level of abstraction as the remainder of the application code

Two abstractions
• Entangled queries
• Entangled transactions

Lots of open research questions
Questions?

http://www.cs.cornell.edu/johannes
johannes@cs.cornell.edu

Thank you: AFOSR, IARPA, NSF, Microsoft, Yahoo!, Google, Amazon, NEC.