Principles of Managing Uncertain Data

Lecture 6: Data Exchange
<table>
<thead>
<tr>
<th></th>
<th>Table of Contents</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Introduction</td>
</tr>
<tr>
<td>2</td>
<td>Schema Mappings</td>
</tr>
<tr>
<td>3</td>
<td>Source/Target Instances</td>
</tr>
<tr>
<td>4</td>
<td>Answering Target Queries</td>
</tr>
<tr>
<td>5</td>
<td>Finding Universal Solutions</td>
</tr>
</tbody>
</table>
Background

- Data exchange: transform data between different applications, each with its own schema
- An old problem (e.g., the EXPRESS system [SHT+77])
- Revisited and formalized by Fagin et al. [FKMP03] in 2003
  - Test-of-time award in ICDT 2013
- Lots of followups since 2013, including specialized sessions in main database conferences
- Implementation by the Clio project (IBM Almaden + University of Toronto), productized within IBM DB2
- Application to various data models (XML, RDF)
The Clio Project (1)
The Clio Project (2)
Data exchange: semantics and query answering

Authors: Ronald Fagin, Phokion G Kolaitis, Renée J Miller, Lucian Popa

Publication date: 2005/5/25

Journal: Theoretical Computer Science

Volume: 336

Issue: 1

Pages: 89-124

Publisher: Elsevier

Description: Data exchange is the problem of taking data structured under a source schema and creating an instance of a target schema that reflects the source data as accurately as possible. In this paper, we address foundational and algorithmic issues related to the semantics of data exchange and to the query answering problem in the context of data exchange. These issues arise because, given a source instance, there may be many target instances that satisfy the constraints of the data exchange problem.

We give an algebraic specification that selects, among all solutions to the data exchange problem, a special class of solutions that we call universal. We show that a universal solution has no more and no less data than required for data exchange and that it represents the entire space of possible solutions. We then identify fairly general, yet practical, conditions that guarantee the existence of a universal solution and yield ...
Popular Research Challenges

- How to materialize target data from source data?
- Answering target queries given source data
  - Semantics, complexity
- Manipulating mapping specifications
  - Composition, inversion
Popular Research Challenges

- How to materialize target data from source data?
- Answering target queries given source data
  - Semantics, complexity
- Manipulating mapping specifications
  - Composition, inversion

The core of these challenges is missing information: underspecified logical rules, missing target attributes
Running Example

Source:

<table>
<thead>
<tr>
<th>ManagedBy</th>
</tr>
</thead>
<tbody>
<tr>
<td>emp</td>
</tr>
<tr>
<td>Barak</td>
</tr>
<tr>
<td>Chen</td>
</tr>
<tr>
<td>Ella</td>
</tr>
</tbody>
</table>

Target:

- No manager manages >1 departments
- Every manager is an employee
Table of Contents

1. Introduction
2. Schema Mappings
3. Source/Target Instances
4. Answering Target Queries
5. Finding Universal Solutions
**Definition (Schema Mapping) [FKMP05]**

A *schema mapping* is a triple \((S, T, \Sigma)\) where:

- \(S = \{S_1, \ldots, S_n\}\) is a *source schema*
- \(T = \{T_1, \ldots, T_m\}\) is a *target schema*
- \(S\) and \(T\) have no common relation names
- \(\Sigma\) is a set of logical formulas over \(S \cup T\)
### Example

| S: ManagedBy | |  
|----------|---|---|  
| emp | mgr |  

| T: Emp & Dept | |  
|----------|---|---|  
| emp | dept | dept | mgr |  

- If \( e \) is managed by \( m \), then \( e \) belongs to a department managed by \( m \)
- \( \Sigma: \) Every manager is an employee of her department
- No manager manages more than one department
We consider three types of constraints in $\Sigma$

- Source-to-target tuple-generating dependencies
  - st-TGDs
- Target tuple-generating dependencies
  - t-TGDs
- Target equality-generating dependencies
  - t-EGDs
Let $(S, T, \Sigma)$ be a schema mapping.
Let \((S, T, \Sigma)\) be a schema mapping

Recall: a TGD is an expression of the form

\[
\forall x [ \varphi(x) \rightarrow \exists y \psi(x, y) ]
\]

where \(\varphi(x)\) and \(\psi(x, y)\) are conjunctions of atomic formulas \(R(\tau_1, \ldots, \tau_k)\)
Let \((S, T, \Sigma)\) be a schema mapping.

Recall: a TGD is an expression of the form

\[
\forall x[\varphi(x) \to \exists y\psi(x, y)]
\]

where \(\varphi(x)\) and \(\psi(x, y)\) are conjunctions of atomic formulas

\(R(\tau_1, \ldots, \tau_k)\)

In an st-TGD, \(\varphi(x)\) is over \(S\) (the source schema), and
\(\psi(x, y)\) is over \(T\) (the target schema).

In a t-TGD, both \(\varphi(x)\) and \(\psi(x, y)\) are over \(T\).
### Example

**S:**

<table>
<thead>
<tr>
<th>ManagedBy</th>
</tr>
</thead>
<tbody>
<tr>
<td>emp</td>
</tr>
</tbody>
</table>

**T:**

<table>
<thead>
<tr>
<th>Emp</th>
<th>Dept</th>
</tr>
</thead>
<tbody>
<tr>
<td>emp</td>
<td>dept</td>
</tr>
<tr>
<td>dept</td>
<td>mgr</td>
</tr>
</tbody>
</table>

**Σ:**

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Example

S: \[\begin{array}{cc}
\text{ManagedBy} \\
\text{emp} & \text{mgr}
\end{array}\]

T: \[\begin{array}{cc}
\text{Emp} \\
\text{emp} & \text{dept} \\
\text{Dept} \\
\text{dept} & \text{mgr}
\end{array}\]

\[\Sigma:\text{st-TGD: ManagedBy}(e, m) \rightarrow \text{Emp}(e, d), \text{Dept}(d, m)\]
Example

\[ \begin{array}{c|c|c}
S: & \text{ManagedBy} & \\
 & \text{emp} & \text{mgr} \\
\hline
T: & \text{Emp} & \text{Dept} & \\
 & \text{emp} & \text{dept} & \text{dept} & \text{mgr} \\
\hline
\Sigma: & \text{st-TGD: ManagedBy}(e, m) \rightarrow \text{Emp}(e, d), \text{Dept}(d, m) & \\
 & \text{t-TGD: Dept}(d, m) \rightarrow \text{Emp}(m, d) & \\
\end{array} \]
Target EGDs

- Let \((S, T, \Sigma)\) be a schema mapping
- Recall that an EGD is an expression of the form

\[
\forall x [ \varphi(x) \rightarrow y_1 = y_2 ]
\]

- \(\varphi(x)\) is a conjunction of atomic formulas
- \(y_1\) and \(y_2\) are variables in \(x\)
Target EGDs

- Let \((S, T, \Sigma)\) be a schema mapping
- Recall that an EGD is an expression of the form

\[
\forall x [\varphi(x) \rightarrow y_1 = y_2]
\]

- \(\varphi(x)\) is a conjunction of atomic formulas
- \(y_1\) and \(y_2\) are variables in \(x\)
- In a \textit{t-EGD}, \(\varphi(x)\) is over the target schema
Let \((S, T, \Sigma)\) be a schema mapping.

Recall that an EGD is an expression of the form

\[
\forall x \left[ \varphi(x) \rightarrow y_1 = y_2 \right]
\]

- \(\varphi(x)\) is a conjunction of atomic formulas
- \(y_1\) and \(y_2\) are variables in \(x\)

In a \(t\)-EGD, \(\varphi(x)\) is over the target schema.

Example: no manager manages more than one department

\[
\text{Dept}(d, m), \text{Dept}(d', m) \rightarrow d = d'
\]
### Complete Example (Schema Mapping)

#### S:

<table>
<thead>
<tr>
<th>ManagedBy</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>emp</td>
<td>mgr</td>
</tr>
</tbody>
</table>

#### T:

<table>
<thead>
<tr>
<th>Emp</th>
<th>Dept</th>
</tr>
</thead>
<tbody>
<tr>
<td>emp</td>
<td>dept</td>
</tr>
<tr>
<td>dept</td>
<td>mgr</td>
</tr>
</tbody>
</table>

#### Constraints

- **st-TGD:** \( ManagedBy(e, m) \rightarrow Emp(e, d), Dept(d, m) \)
- **Σ:** \( t-TGD: Dept(d, m) \rightarrow Emp(m, d) \)
- **t-EGD:** \( Dept(d, m), Dept(d', m) \rightarrow d = d' \)
<table>
<thead>
<tr>
<th></th>
<th>Table of Contents</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Introduction</td>
</tr>
<tr>
<td>2</td>
<td>Schema Mappings</td>
</tr>
<tr>
<td>3</td>
<td>Source/Target Instances</td>
</tr>
<tr>
<td>4</td>
<td>Answering Target Queries</td>
</tr>
<tr>
<td>5</td>
<td>Finding Universal Solutions</td>
</tr>
</tbody>
</table>
Let \((S, T, \Sigma)\) be a schema mapping

- A **source instance** is an instance over \(S\)
- A **target instance** is a v-instance \(J\) over \(T\); hence, \(J\) may have **labeled nulls** (or **variables**) instead of values
Let \((S, T, \Sigma)\) be a schema mapping. A source instance is an instance over \(S\), a target instance is a \(v\)-instance \(J\) over \(T\); hence, \(J\) may have labeled nulls (or variables) instead of values. Formally:

- We have a set \(\text{Const}\) of constants and a set \(\text{Var}\) of variables (labeled nulls).
- A source instance has only values from \(\text{Const}\)
- A target instance may have values from both \(\text{Const}\) and \(\text{Var}\)
For the sake of constraint satisfaction, we identify a target instance as an ordinary instance, interpreting each labeled null as a unique value.
For the sake of constraint satisfaction, we identify a target instance as an ordinary instance, interpreting each labeled null as a unique value.

This is important, since we can now define what it means for a target instance (with nulls) to satisfy a TGD/EGD.
**Definition (Solution)**

Let $\mathcal{M} = (S, T, \Sigma)$ be a schema mapping, and let $I$ be a source instance. A *solution* is a target instance $J$, such that $I$ and $J$ jointly satisfy $\Sigma$. We denote by $Sol_{\mathcal{M}}(I)$ the set of all solutions.
Example

<table>
<thead>
<tr>
<th>ManagedBy</th>
</tr>
</thead>
<tbody>
<tr>
<td>emp</td>
</tr>
<tr>
<td>Barak</td>
</tr>
<tr>
<td>Chen</td>
</tr>
<tr>
<td>Ella</td>
</tr>
</tbody>
</table>

- ManagedBy(e, m) → Emp(e, d), Dept(d, m)
- Dept(d, m) → Emp(m, d)
- Dept(d, m), Dept(d', m) → d = d'
### Example

**ManagedBy**

<table>
<thead>
<tr>
<th>emp</th>
<th>mgr</th>
</tr>
</thead>
<tbody>
<tr>
<td>Barak</td>
<td>Ahuva</td>
</tr>
<tr>
<td>Chen</td>
<td>Ahuva</td>
</tr>
<tr>
<td>Ella</td>
<td>Doron</td>
</tr>
</tbody>
</table>

- ManagedBy\((e, m)\) → Emp\((e, d)\), Dept\((d, m)\)
- Dept\((d, m)\) → Emp\((m, d)\)
- Dept\((d, m)\), Dept\((d', m)\) → \(d = d'\)

**Emp**

<table>
<thead>
<tr>
<th>emp</th>
<th>dept</th>
</tr>
</thead>
<tbody>
<tr>
<td>Barak</td>
<td>R&amp;D</td>
</tr>
<tr>
<td>Chen</td>
<td>R&amp;D</td>
</tr>
<tr>
<td>Ella</td>
<td>Finance</td>
</tr>
<tr>
<td>Ahuva</td>
<td>R&amp;D</td>
</tr>
<tr>
<td>Doron</td>
<td>Finance</td>
</tr>
</tbody>
</table>

**Dept**

<table>
<thead>
<tr>
<th>dept</th>
<th>mgr</th>
</tr>
</thead>
<tbody>
<tr>
<td>R&amp;D</td>
<td>Ahuva</td>
</tr>
<tr>
<td>Finance</td>
<td>Doron</td>
</tr>
</tbody>
</table>
### Example

<table>
<thead>
<tr>
<th>ManagedBy</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>emp</td>
<td>mgr</td>
</tr>
<tr>
<td>Barak</td>
<td>Ahuva</td>
</tr>
<tr>
<td>Chen</td>
<td>Ahuva</td>
</tr>
<tr>
<td>Ella</td>
<td>Doron</td>
</tr>
</tbody>
</table>

- ManagedBy($e, m$) $\rightarrow$ Emp($e, d$), Dept($d, m$)
- Dept($d, m$) $\rightarrow$ Emp($m, d$)
- Dept($d, m$), Dept($d', m$) $\rightarrow d = d'$

<table>
<thead>
<tr>
<th>Emp</th>
<th>Dept</th>
</tr>
</thead>
<tbody>
<tr>
<td>emp</td>
<td>dept</td>
</tr>
<tr>
<td>Barak</td>
<td>$\bot_1$</td>
</tr>
<tr>
<td>Chen</td>
<td>$\bot_1$</td>
</tr>
<tr>
<td>Ella</td>
<td>Finance</td>
</tr>
<tr>
<td>Ahuva</td>
<td>$\bot_1$</td>
</tr>
<tr>
<td>Doron</td>
<td>Finance</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>dept</th>
<th>mgr</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\bot_1$</td>
<td>Ahuva</td>
</tr>
<tr>
<td>Finance</td>
<td>Doron</td>
</tr>
</tbody>
</table>
### Example

**ManagedBy**

<table>
<thead>
<tr>
<th>emp</th>
<th>mgr</th>
</tr>
</thead>
<tbody>
<tr>
<td>Barak</td>
<td>Ahuva</td>
</tr>
<tr>
<td>Chen</td>
<td>Ahuva</td>
</tr>
<tr>
<td>Ella</td>
<td>Doron</td>
</tr>
</tbody>
</table>

- $\text{ManagedBy}(e, m) \rightarrow \text{Emp}(e, d), \text{Dept}(d, m)$
- $\text{Dept}(d, m) \rightarrow \text{Emp}(m, d)$
- $\text{Dept}(d, m), \text{Dept}(d', m) \rightarrow d = d'$

**Emp**

<table>
<thead>
<tr>
<th>emp</th>
<th>dept</th>
</tr>
</thead>
<tbody>
<tr>
<td>Barak</td>
<td>$\perp_1$</td>
</tr>
<tr>
<td>Chen</td>
<td>$\perp_1$</td>
</tr>
<tr>
<td>Ella</td>
<td>$\perp_2$</td>
</tr>
<tr>
<td>Ahuva</td>
<td>$\perp_1$</td>
</tr>
<tr>
<td>Doron</td>
<td>$\perp_2$</td>
</tr>
</tbody>
</table>

**Dept**

<table>
<thead>
<tr>
<th>dept</th>
<th>mgr</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\perp_1$</td>
<td>Ahuva</td>
</tr>
<tr>
<td>$\perp_2$</td>
<td>Doron</td>
</tr>
</tbody>
</table>
## Example

**ManagedBy**

<table>
<thead>
<tr>
<th>emp</th>
<th>mgr</th>
</tr>
</thead>
<tbody>
<tr>
<td>Barak</td>
<td>Ahuva</td>
</tr>
<tr>
<td>Chen</td>
<td>Ahuva</td>
</tr>
<tr>
<td>Ella</td>
<td>Doron</td>
</tr>
</tbody>
</table>

- ManagedBy\((e, m)\) → Emp\((e, d)\), Dept\((d, m)\)
- Dept\((d, m)\) → Emp\((m, d)\)
- Dept\((d, m)\), Dept\((d', m)\) → \(d = d'\)

**Emp**

<table>
<thead>
<tr>
<th>emp</th>
<th>dept</th>
</tr>
</thead>
<tbody>
<tr>
<td>Barak</td>
<td>⊥₁</td>
</tr>
<tr>
<td>Chen</td>
<td>⊥₁</td>
</tr>
<tr>
<td>Ella</td>
<td>⊥₁</td>
</tr>
<tr>
<td>Ahuva</td>
<td>⊥₁</td>
</tr>
<tr>
<td>Doron</td>
<td>⊥₁</td>
</tr>
</tbody>
</table>

**Dept**

<table>
<thead>
<tr>
<th>dept</th>
<th>mgr</th>
</tr>
</thead>
<tbody>
<tr>
<td>⊥₁</td>
<td>Ahuva</td>
</tr>
<tr>
<td>⊥₁</td>
<td>Doron</td>
</tr>
</tbody>
</table>
Homomorphism

- Let $J$ and $K$ be two target instances
Homomorphism

- Let $J$ and $K$ be two target instances
- Let $\mu$ be a mapping from the values of $J$ to those of $K$
Let $J$ and $K$ be two target instances
Let $\mu$ be a mapping from the values of $J$ to those of $K$
For a fact $f = R(a_1, \ldots, a_k)$, we denote the fact $R(\mu(a_1), \ldots, \mu(a_k))$ by $\mu(f)$
Let $J$ and $K$ be two target instances

Let $\mu$ be a mapping from the values of $J$ to those of $K$

For a fact $f = R(a_1, \ldots, a_k)$, we denote the fact $R(\mu(a_1), \ldots, \mu(a_k))$ by $\mu(f)$

$\mu$ is a **homomorphism** from $J$ to $K$ if

1. $\mu(c) = c$ for every $c \in \text{Const}$; and
2. $\mu(f) \in K$ for every $f \in J$. 
Homomorphism

- Let $J$ and $K$ be two target instances
- Let $\mu$ be a mapping from the values of $J$ to those of $K$
- For a fact $f = R(a_1, \ldots, a_k)$, we denote the fact $R(\mu(a_1), \ldots, \mu(a_k))$ by $\mu(f)$
- $\mu$ is a **homomorphism** from $J$ to $K$ if
  1. $\mu(c) = c$ for every $c \in \text{Const}$; and
  2. $\mu(f) \in K$ for every $f \in J$.
- That is, a homomorphism maps every constant to itself, and every $J$-fact to a $K$-fact
Let $J$ and $K$ be two target instances

We denote by $J \rightarrow K$ the fact that there exists a homomorphism from $J$ to $K$
Let $J$ and $K$ be two target instances

- We denote by $J \rightarrow K$ the fact that there exists a homomorphism from $J$ to $K$

- Note: many homomorphisms from $J$ to $K$ may exist
Let $J$ and $K$ be two target instances.

- We denote by $J \rightarrow K$ the fact that there exists a homomorphism from $J$ to $K$.
- Note: many homomorphisms from $J$ to $K$ may exist.
- $J \rightarrow K$ means that $K$ contains a homomorphic image of $J$, that is, a set of facts that could be obtained from $J$ by an assignment (of constants or variables) to the variables.
Example

\[ \begin{array}{c|c}
\text{Emp} & \\
\hline
\text{emp} & \text{dept} \\
Ahuva & \bot_1 \\
Barak & \bot_2 \\
Chen & \text{Eng} \\
Chen & \bot_3 \\
\bot_4 & \bot_5 \\
\end{array} \quad \rightarrow \quad \begin{array}{c|c}
\text{Emp} & \\
\hline
\text{emp} & \text{dept} \\
Ahuva & \bot_1 \\
Barak & \bot_1 \\
Chen & \text{Eng} \\
\end{array} \]

\[ \begin{array}{c|c}
\text{Dept} & \\
\hline
\text{dept} & \text{mgr} \\
\bot_1 & \bot_4 \\
\bot_2 & Ahuva \\
\text{Eng} & Fady \\
\bot_6 & Barak \\
\end{array} \quad \rightarrow \quad \begin{array}{c|c}
\text{Dept} & \\
\hline
\text{dept} & \text{mgr} \\
\bot_1 & Ahuva \\
\text{Eng} & Fady \\
\text{Finance} & Barak \\
\bot_2 & Doron \\
\end{array} \]
### Example

<table>
<thead>
<tr>
<th>Dept</th>
<th>dept</th>
<th>mgr</th>
</tr>
</thead>
<tbody>
<tr>
<td>dept</td>
<td>mgr</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>dept</td>
<td>mgr</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

\[ \mu : \]

\[
\begin{align*}
\bot_1 & \rightarrow \bot_1 \\
\bot_2 & \rightarrow \bot_1 \\
\bot_3 & \rightarrow \text{Eng} \\
\bot_4 & \rightarrow \text{Ahuva} \\
\bot_5 & \rightarrow \bot_1 \\
\bot_6 & \rightarrow \text{Finance} \\
\end{align*}
\]

Ahuva \rightarrow Ahuva

...
Properties of Homomorphism

- The relationship $\rightarrow$ is reflexive
  - $J \rightarrow J$ for all $J$

- The relationship $\rightarrow$ is transitive
  - For all $J$, $K$ and $L$, if $J \rightarrow K$ and $K \rightarrow L$ then $J \rightarrow L$
Properties of Homomorphism

- The relationship \( \rightarrow \) is **reflexive**
  - \( J \rightarrow J \) for all \( J \)

- The relationship \( \rightarrow \) is **transitive**
  - For all \( J, K \) and \( L \), if \( J \rightarrow K \) and \( K \rightarrow L \) then \( J \rightarrow L \)

- **Why?**
Properties of Homomorphism

- The relationship $\rightarrow$ is reflexive
  - $J \rightarrow J$ for all $J$

- The relationship $\rightarrow$ is transitive
  - For all $J$, $K$ and $L$, if $J \rightarrow K$ and $K \rightarrow L$ then $J \rightarrow L$
  - Why?

- The relationship $\rightarrow$ is not symmetric
Properties of Homomorphism

- The relationship $\rightarrow$ is reflexive
  - $J \rightarrow J$ for all $J$

- The relationship $\rightarrow$ is transitive
  - For all $J$, $K$ and $L$, if $J \rightarrow K$ and $K \rightarrow L$ then $J \rightarrow L$

  Why?

- The relationship $\rightarrow$ is not symmetric
  - Example?
Isomorphism

- Two target instances \( J \) and \( K \) are *isomorphic* if each one can be obtained from the other by (uniquely) renaming variables.
- Isomorphism is an *equivalence* relation (i.e., reflexive, symmetric and transitive).
Isomorphism

- Two target instances $J$ and $K$ are *isomorphic* if each one can be obtained from the other by (uniquely) renaming variables.
- Isomorphism is an *equivalence* relation (i.e., reflexive, symmetric, and transitive).
- Clearly, if $J$ and $K$ are isomorphic, then $J \rightarrow K$ and $K \rightarrow J$.
  - But not vice versa.
**Universal Solutions**

**Definition (Universal Solution)**

Let $\mathcal{M} = (S, T, \Sigma)$ be a schema mapping, and let $I$ be a source instance. A *universal solution* is a solution $J$ such that $J \rightarrow K$ for all solutions $K$. 

---

**Instances**

**Solutions**

**Homomorphism**

**Universal Solutions**

**Core Solutions**

**References**
Example of a Universal Solution

<table>
<thead>
<tr>
<th>ManagedBy</th>
</tr>
</thead>
<tbody>
<tr>
<td>emp</td>
</tr>
<tr>
<td>Barak</td>
</tr>
<tr>
<td>Chen</td>
</tr>
<tr>
<td>Ella</td>
</tr>
</tbody>
</table>

- ManagedBy\((e, m)\) $\rightarrow$ Emp\((e, d)\), Dept\((d, m)\)
- Dept\((d, m)\) $\rightarrow$ Emp\((m, d)\)
- Dept\((d, m)\), Dept\((d', m)\) $\rightarrow$ $d = d'$

Later: how to prove that this is indeed a universal solution?
Example of a Universal Solution

**ManagedBy**

<table>
<thead>
<tr>
<th>emp</th>
<th>mgr</th>
</tr>
</thead>
<tbody>
<tr>
<td>Barak</td>
<td>Ahuva</td>
</tr>
<tr>
<td>Chen</td>
<td>Ahuva</td>
</tr>
<tr>
<td>Ella</td>
<td>Doron</td>
</tr>
</tbody>
</table>

- ManagedBy$(e, m) \rightarrow \text{Emp}(e, d), \text{Dept}(d, m)$
- Dept$(d, m) \rightarrow \text{Emp}(m, d)$
- Dept$(d, m), \text{Dept}(d', m) \rightarrow d = d'$

**Emp**

<table>
<thead>
<tr>
<th>emp</th>
<th>dept</th>
</tr>
</thead>
<tbody>
<tr>
<td>Barak</td>
<td>$\bot_1$</td>
</tr>
<tr>
<td>Chen</td>
<td>$\bot_1$</td>
</tr>
<tr>
<td>Ella</td>
<td>$\bot_2$</td>
</tr>
<tr>
<td>Ahuva</td>
<td>$\bot_1$</td>
</tr>
<tr>
<td>Doron</td>
<td>$\bot_2$</td>
</tr>
</tbody>
</table>

**Dept**

<table>
<thead>
<tr>
<th>dept</th>
<th>mgr</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\bot_1$</td>
<td>Ahuva</td>
</tr>
<tr>
<td>$\bot_2$</td>
<td>Doron</td>
</tr>
</tbody>
</table>
Example of a Universal Solution

**ManagedBy**

<table>
<thead>
<tr>
<th>emp</th>
<th>mgr</th>
</tr>
</thead>
<tbody>
<tr>
<td>Barak</td>
<td>Ahuva</td>
</tr>
<tr>
<td>Chen</td>
<td>Ahuva</td>
</tr>
<tr>
<td>Ella</td>
<td>Doron</td>
</tr>
</tbody>
</table>

- ManagedBy(e, m) $\rightarrow$ Emp(e, d), Dept(d, m)
- Dept(d, m) $\rightarrow$ Emp(m, d)
- Dept(d, m), Dept(d', m) $\rightarrow$ d = d'

**Emp**

<table>
<thead>
<tr>
<th>emp</th>
<th>dept</th>
</tr>
</thead>
<tbody>
<tr>
<td>Barak</td>
<td>$\bot_1$</td>
</tr>
<tr>
<td>Chen</td>
<td>$\bot_1$</td>
</tr>
<tr>
<td>Ella</td>
<td>$\bot_2$</td>
</tr>
<tr>
<td>Ahuva</td>
<td>$\bot_1$</td>
</tr>
<tr>
<td>Doron</td>
<td>$\bot_2$</td>
</tr>
</tbody>
</table>

**Dept**

<table>
<thead>
<tr>
<th>dept</th>
<th>mgr</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\bot_1$</td>
<td>Ahuva</td>
</tr>
<tr>
<td>$\bot_2$</td>
<td>Doron</td>
</tr>
<tr>
<td>$\bot_3$</td>
<td>$\bot_3$</td>
</tr>
</tbody>
</table>
Example of a Universal Solution

### ManagedBy

<table>
<thead>
<tr>
<th>emp</th>
<th>mgr</th>
</tr>
</thead>
<tbody>
<tr>
<td>Barak</td>
<td>Ahuva</td>
</tr>
<tr>
<td>Chen</td>
<td>Ahuva</td>
</tr>
<tr>
<td>Ella</td>
<td>Doron</td>
</tr>
</tbody>
</table>

- ManagedBy($e, m$) $\rightarrow$ Emp($e, d$), Dept($d, m$)
- Dept($d, m$) $\rightarrow$ Emp($m, d$)
- Dept($d, m$), Dept($d', m$) $\rightarrow$ $d = d'$

### Emp

<table>
<thead>
<tr>
<th>emp</th>
<th>dept</th>
</tr>
</thead>
<tbody>
<tr>
<td>Barak</td>
<td>$\bot_1$</td>
</tr>
<tr>
<td>Chen</td>
<td>$\bot_1$</td>
</tr>
<tr>
<td>Ella</td>
<td>$\bot_2$</td>
</tr>
<tr>
<td>Ahuva</td>
<td>$\bot_1$</td>
</tr>
<tr>
<td>Doron</td>
<td>$\bot_2$</td>
</tr>
</tbody>
</table>

### Dept

<table>
<thead>
<tr>
<th>dept</th>
<th>mgr</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\bot_1$</td>
<td>Ahuva</td>
</tr>
<tr>
<td>$\bot_2$</td>
<td>Doron</td>
</tr>
<tr>
<td>$\bot_2$</td>
<td>$\bot_3$</td>
</tr>
</tbody>
</table>

Later: how to prove that this is indeed a universal solution?
Why Universal Solutions?

- If $J$ is a universal solution, then every possible solution contains a homomorphic image of $J$.
- Intuitively, it means that $J$ does not contain facts that are not justified (in some strong sense).
Why Universal Solutions?

- If $J$ is a universal solution, then every possible solution contains a homomorphic image of $J$.
- Intuitively, it means that $J$ does not contain facts that are not justified (in some strong sense).
- Universal solutions are important for several reasons:
Why Universal Solutions?

- If $J$ is a universal solution, then every possible solution contains a homomorphic image of $J$.
- Intuitively, it means that $J$ does not contain facts that are not justified (in some strong sense).
- Universal solutions are important for several reasons:
  - They are “legitimate” solutions in the sense that whatever they contain can be found (through homomorphism) in every possible solution.
Why Universal Solutions?

- If $J$ is a universal solution, then every possible solution contains a homomorphic image of $J$.
- Intuitively, it means that $J$ does not contain facts that are not justified (in some strong sense).
- Universal solutions are important for several reasons:
  - They are “legitimate” solutions in the sense that whatever they contain can be found (through homomorphism) in every possible solution.
  - Under fairly general assumptions, we can actually materialize (i.e., produce) a universal solution.
Why Universal Solutions?

- If $J$ is a universal solution, then every possible solution contains a homomorphic image of $J$.
- Intuitively, it means that $J$ does not contain facts that are not justified (in some strong sense).
- Universal solutions are important for several reasons:
  - They are “legitimate” solutions in the sense that whatever they contain can be found (through homomorphism) in every possible solution.
  - Under fairly general assumptions, we can actually materialize (i.e., produce) a universal solution.
  - They can be used for answering positive RA queries (later).
Variety of Universal Solutions

- As we have seen, universal solutions are not necessarily isomorphic to each other
- Moreover, different universal solutions can have different sizes
Variety of Universal Solutions

- As we have seen, universal solutions are not necessarily isomorphic to each other.
- Moreover, different universal solutions can have different sizes.
- They are, though, *homomorphically equivalent* to one another.
  - That is, there are homomorphisms in both directions.
Cores of Instances

- Let \( J \) be a \( v \)-instance

- An *endomorphism* on \( J \) is a homomorphism from \( J \) to itself
Cores of Instances

- Let $J$ be a v-instance.
- An \textit{endomorphism} on $J$ is a homomorphism from $J$ to itself.
- The endomorphism is \textit{proper} if it maps $J$ to a proper subset of $J$. 
Let $J$ be a $v$-instance

- An *endomorphism* on $J$ is a homomorphism from $J$ to itself
- The endomorphism is *proper* if it maps $J$ to a proper subset of $J$
- The *core* of $J$, denoted $\text{Core}(J)$, is a subinstance $J'$ of $J$ such that:
  - There is a homomorphism (endomorphism) from $J$ to $J'$
  - There is no proper endomorphism on $J'$
Let $J$ be a $v$-instance

- An *endomorphism* on $J$ is a homomorphism from $J$ to itself
- The endomorphism is *proper* if it maps $J$ to a proper subset of $J$
- The *core* of $J$, denoted $\text{Core}(J)$, is a subinstance $J'$ of $J$ such that:
  - There is a homomorphism (endomorphism) from $J$ to $J'$
  - There is no proper endomorphism on $J'$
- Why “the” core? Next...
**Theorem [HN92]**

The following hold:

1. Every v-instance has a core.
2. All cores of a v-instance are isomorphic.
3. If $J \rightarrow K$ and $K \rightarrow J$, then $\text{Core}(J) \cong \text{Core}(K)$.
**Theorem [HN92]**

The following hold:

- Every $v$-instance has a core.
Fundamental Theorem

**Theorem [HN92]**

The following hold:

- Every $v$-instance has a core.
- All cores of a $v$-instance are isomorphic.
Theorem [HN92]

The following hold:

- Every $v$-instance has a core.
- All cores of a $v$-instance are isomorphic.
- If $J \rightarrow K$ and $K \rightarrow J$, then $\text{Core}(J) \cong \text{Core}(K)$. 
Example 1

\[ T \]

\begin{array}{c|c}
\hline
a_1 & a_2 \\
\hline
\top_1 & a \\
a & a \\
a & \top_2 \\
\top_1 & \top_2 \\
\top_2 & \top_3 \\
\hline
\end{array}

\[ \Rightarrow \]

\[ T \]

\begin{array}{c|c}
\hline
\top_1 & \top_1 \\
\top_2 & a \\
\top_3 & \top_2 \\
\hline
\end{array}

Example 2

\[
\begin{array}{c|c}
T & \Rightarrow \\
\hline
a_1 & a_2 \\
\hline
\bot_1 & a \\
a & a \\
a & \bot_2 \\
\bot_1 & \bot_2 \\
\bot_2 & \bot_3 \\
\end{array}
\quad
\Rightarrow
\quad
\begin{array}{c|c}
T & \\
\hline
a_1 & a_2 \\
\hline
\bot_1 & a \\
\bot_2 & a \\
a & a \\
\end{array}
\]
Endomorphic Images of Solutions

- Let \((S, T, \Sigma)\) be a schema mapping, and let \(I\) be a source instance.
- Suppose that \(J\) is a solution and \(\mu\) is an endomorphism over \(J\).
- \(\mu(J)\) necessarily a solution?
Endomorphic Images of Solutions

- Let \((S, T, \Sigma)\) be a schema mapping, and let \(I\) be a source instance.
- Suppose that \(J\) is a solution and \(\mu\) is an endomorphism over \(J\).
- \(\mu(J)\) necessarily a solution?
- No!
Example 1 Revisited

\[ R(x) \rightarrow T(x, x) \]
\[ T(x, y), T(y, y), T(y, z) \rightarrow T(x, z) \]
\[ I = \{ R(a) \} \]

<table>
<thead>
<tr>
<th></th>
<th>( T )</th>
<th>( \Rightarrow )</th>
<th>( T )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( a_1 )</td>
<td>( a_2 )</td>
<td>( \bot_1 \leftrightarrow \bot_1 )</td>
<td>( a_1 )</td>
</tr>
<tr>
<td>( \bot_1 )</td>
<td>a</td>
<td>( \bot_2 \leftrightarrow a )</td>
<td>( \bot_1 )</td>
</tr>
<tr>
<td>a</td>
<td>a</td>
<td>( \bot_3 \leftrightarrow \bot_2 )</td>
<td>( a )</td>
</tr>
<tr>
<td>a</td>
<td>( \bot_2 )</td>
<td></td>
<td>( a )</td>
</tr>
<tr>
<td>( \bot_1 )</td>
<td>( \bot_2 )</td>
<td></td>
<td>( a )</td>
</tr>
<tr>
<td>( \bot_2 )</td>
<td>( \bot_3 )</td>
<td></td>
<td>( \bot_2 )</td>
</tr>
</tbody>
</table>
**Theorem [FKP05]**

Let \((S, T, \Sigma)\) be a schema mapping, let \(I\) be a source instance, and let \(J\) be target instance. If \(J\) is a solution, then \(\text{Core}(J)\) is a solution.
We now get a good candidate for a solution: the core of a universal solution

- Simply called a core solution
Cores of Universal Solutions [FKP05]

- We now get a good candidate for a solution: the core of a universal solution
  - Simply called a *core solution*
- We know:
  - There is a universal solution if and only if there is a core solution
  - All core solutions are isomorphic
  - Core solutions are universal solutions
We now get a good candidate for a solution: the core of a universal solution

Simply called a core solution

We know:

There is a universal solution if and only if there is a core solution
We now get a good candidate for a solution: the core of a universal solution

- Simply called a *core solution*

We know:

- There is a universal solution if and only if there is a core solution
- All core solutions are isomorphic
We now get a good candidate for a solution: the core of a universal solution

- Simply called a \textit{core solution}

We know:

- There is a universal solution if and only if there is a core solution
- All core solutions are isomorphic
- Core solutions are \textit{universal} solutions
  - \textit{why?}
Cores of Universal Solutions [FKP05]

- We now get a good candidate for a solution: the core of a universal solution
  - Simply called a \textit{core solution}
- We know:
  - There is a universal solution if and only if there is a core solution
  - All core solutions are isomorphic
  - Core solutions are \textit{universal} solutions
    - \textit{why}?
  - Core solutions are the \textit{smallest} universal solutions
    - \textit{why}?
We evaluate a query $Q$ over a target instance $J$ by treating each labeled null as a unique value.
We evaluate a query $Q$ over a target instance $J$ by treating each labeled null as a unique value.

Clearly, different solutions $J \in Sol_M(I)$ may have different results $Q(J)$. 
**Definition (Certain Answers)**

Let \( \mathcal{M} \) be a schema mapping, \( I \) a source instance, and \( Q \) be a target query. We denote by \( \text{Certain}_{Q,\mathcal{M}}(I) \) the relation that consists of all the tuples that occur in \( Q(J) \) for all solutions \( J \);
**Definition** (Certain Answers)

Let $\mathcal{M}$ be a schema mapping, $I$ a source instance, and $Q$ be a target query. We denote by $\text{Certain}_{Q,\mathcal{M}}(I)$ the relation that consists of all the tuples that occur in $Q(J)$ for all solutions $J$; that is:

$$\text{Certain}_{Q,\mathcal{M}}(I) \overset{\text{def}}{=} \cap\{Q(J) \mid J \in \text{Sol}_{\mathcal{M}}(I)\}$$
**Definition (Certain Answers)**

Let $\mathcal{M}$ be a schema mapping, $I$ a source instance, and $Q$ be a target query. We denote by $\text{Certain}_{Q,\mathcal{M}}(I)$ the relation that consists of all the tuples that occur in $Q(J)$ for all solutions $J$; that is:

$$
\text{Certain}_{Q,\mathcal{M}}(I) \overset{\text{def}}{=} \cap \{Q(J) \mid J \in \text{Sol}_{\mathcal{M}}(I)\}
$$

A tuple in $\text{Certain}_{Q,\mathcal{M}}(I)$ is called a *certain answer*. 

Answering UCQs

- Recall that a conjunctive query (CQ) is a query of the form

\[ Q(x) : \exists y [\varphi_1(x, y) \land \cdots \land \varphi_k(x, y)] \]

where each \( \varphi_k(x, y) \) is an atomic query
Recall that a conjunctive query (CQ) is a query of the form

\[ Q(\mathbf{x}) :\exists \mathbf{y}[\varphi_1(\mathbf{x}, \mathbf{y}) \land \cdots \land \varphi_k(\mathbf{x}, \mathbf{y})] \]

where each \( \varphi_k(\mathbf{x}, \mathbf{y}) \) is an atomic query

A union of conjunctive queries (UCQ) is a query of the form

\[ Q(\mathbf{x}) :\neg \psi_1(\mathbf{x}) \lor \cdots \lor \psi_k(\mathbf{x}) \]

where each \( Q(\mathbf{x}) :\neg \psi_i(\mathbf{x}) \) is a CQ
Answering UCQs

- Recall that a conjunctive query (CQ) is a query of the form

\[
Q(x) :\neg \exists y[\varphi_1(x, y) \land \cdots \land \varphi_k(x, y)]
\]

where each \(\varphi_k(x, y)\) is an atomic query

- A union of conjunctive queries (UCQ) is a query of the form

\[
Q(x) :\neg \psi_1(x) \lor \cdots \lor \psi_k(x)
\]

where each \(Q(x) :\neg \psi_i(x)\) is a CQ

- If \(Q\) is a target query and \(J\) is a target instance, then \(Q(J)\downarrow\) denotes the relation obtained from \(Q(J)\) by removing every tuple with one or more nulls
Answering UCQs via Universal Solutions

**Theorem (FKMP’03)**

Let $\mathcal{M} = (S, T, \Sigma)$ be a schema mapping, $Q$ a target UCQ, and $I$ a source instance. If $J$ is a universal solution, then

$$Q(J)\downarrow = \text{Certain}_{Q,\mathcal{M}}(I).$$
The chase is a standard technique in reasoning about databases with integrity constraints [BV84].
The *chase* is a standard technique in reasoning about databases with integrity constraints [BV84].

**Idea:**
- Start with the original database instance
- Repeatedly, look for a violation of a constraint, and fix by revising the instance
- ... until no violations remain (or the violation cannot be fixed)
Chase Steps for Data Exchange

- Start with $J = \emptyset$
Chase Steps for Data Exchange

- Start with $J = \emptyset$
- TGD violation: $\varphi(x) \rightarrow \exists y \psi(x, y)$
  - **Violation**: an assignment $\alpha$ to $x$, such that $\varphi(\alpha(x))$ holds, but there is no $b$ such that $\psi(\alpha(x), b)$
Chase Steps for Data Exchange

- Start with $J = \emptyset$
- TGD violation: $\varphi(x) \rightarrow \exists y \psi(x, y)$
  - **Violation**: an assignment $\alpha$ to $x$, such that $\varphi(\alpha(x))$ holds, but there is no $b$ such that $\psi(\alpha(x), b)$
  - **Fix**: add the facts of $\psi(\alpha(x), y)$, where each existential variable in $y$ is replaced with a fresh null
Chase Steps for Data Exchange

- Start with $J = \emptyset$
- TGD violation: $\varphi(x) \rightarrow \exists y \psi(x, y)$
  - **Violation**: an assignment $\alpha$ to $x$, such that $\varphi(\alpha(x))$ holds, but there is no $b$ such that $\psi(\alpha(x), b)$
  - **Fix**: add the facts of $\psi(\alpha(x), y)$, where each existential variable in $y$ is replaced with a fresh null
- EGD violation: $\varphi(x) \rightarrow y_1 = y_2$
  - **Violation**: An assignment $\alpha$ to $x$, such that $\alpha(\varphi(x))$ holds, but $\alpha(y_1) \neq \alpha(y_2)$
Chase Steps for Data Exchange

- **Start with** $J = \emptyset$
- **TGD violation:** $\varphi(x) \rightarrow \exists y \psi(x, y)$
  - **Violation:** an assignment $\alpha$ to $x$, such that $\varphi(\alpha(x))$ holds, but there is no $b$ such that $\psi(\alpha(x), b)$
  - **Fix:** add the facts of $\psi(\alpha(x), y)$, where each existential variable in $y$ is replaced with a fresh null
- **EGD violation:** $\varphi(x) \rightarrow y_1 = y_2$
  - **Violation:** An assignment $\alpha$ to $x$, such that $\alpha(\varphi(x))$ holds, but $\alpha(y_1) \neq \alpha(y_2)$
  - **Fix:** 3 cases
Chase Steps for Data Exchange

- **Start with** $J = \emptyset$
- **TGD violation:** $\varphi(x) \rightarrow \exists y \psi(x, y)$
  - **Violation:** an assignment $\alpha$ to $x$, such that $\varphi(\alpha(x))$ holds, but there is no $b$ such that $\psi(\alpha(x), b)$
  - **Fix:** add the facts of $\psi(\alpha(x), y)$, where each existential variable in $y$ is replaced with a fresh null
- **EGD violation:** $\varphi(x) \rightarrow y_1 = y_2$
  - **Violation:** An assignment $\alpha$ to $x$, such that $\alpha(\varphi(x))$ holds, but $\alpha(y_1) \neq \alpha(y_2)$
  - **Fix:** 3 cases
    1. $\alpha(y_1)$ is a labeled null: replace every $\alpha(y_1)$ with $\alpha(y_2)$ in $J$
Chase Steps for Data Exchange

- Start with $J = \emptyset$
- TGD violation: $\varphi(x) \rightarrow \exists y \psi(x, y)$
  - **Violation**: an assignment $\alpha$ to $x$, such that $\varphi(\alpha(x))$ holds, but there is no $b$ such that $\psi(\alpha(x), b)$
  - **Fix**: add the facts of $\psi(\alpha(x), y)$, where each existential variable in $y$ is replaced with a fresh null
- EGD violation: $\varphi(x) \rightarrow y_1 = y_2$
  - **Violation**: An assignment $\alpha$ to $x$, such that $\alpha(\varphi(x))$ holds, but $\alpha(y_1) \neq \alpha(y_2)$
  - **Fix**: 3 cases
    1. $\alpha(y_1)$ is a labeled null: replace every $\alpha(y_1)$ with $\alpha(y_2)$ in $J$
    2. $\alpha(y_2)$ is a labeled null: replace every $\alpha(y_2)$ with $\alpha(y_1)$ in $J$
Chase Steps for Data Exchange

- Start with $J = \emptyset$

- TGD violation: $\varphi(x) \rightarrow \exists y \psi(x, y)$
  - **Violation**: an assignment $\alpha$ to $x$, such that $\varphi(\alpha(x))$ holds, but there is no $b$ such that $\psi(\alpha(x), b)$
  - **Fix**: add the facts of $\psi(\alpha(x), y)$, where each existential variable in $y$ is replaced with a fresh null

- EGD violation: $\varphi(x) \rightarrow y_1 = y_2$
  - **Violation**: An assignment $\alpha$ to $x$, such that $\alpha(\varphi(x))$ holds, but $\alpha(y_1) \neq \alpha(y_2)$
  - **Fix**: 3 cases
    1. $\alpha(y_1)$ is a labeled null: replace every $\alpha(y_1)$ with $\alpha(y_2)$ in $J$
    2. $\alpha(y_2)$ is a labeled null: replace every $\alpha(y_2)$ with $\alpha(y_1)$ in $J$
    3. Both $\alpha(y_1)$ and $\alpha(y_2)$ are constants: terminate with failure
### Chase Example

<table>
<thead>
<tr>
<th>ManagedBy</th>
<th>Emp</th>
<th>Dept</th>
</tr>
</thead>
<tbody>
<tr>
<td>emp</td>
<td>mgr</td>
<td></td>
</tr>
<tr>
<td>Barak</td>
<td>Ahuva</td>
<td></td>
</tr>
<tr>
<td>Chen</td>
<td>Ahuva</td>
<td></td>
</tr>
<tr>
<td>Ella</td>
<td>Doron</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>emp</th>
<th>dept</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>dept</th>
<th>mgr</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

1. ManagedBy(e, m) $\rightarrow$ Emp(e, d), Dept(d, m)
2. Dept(d, m), Dept(d', m) $\rightarrow$ d = d'
3. Dept(d, m) $\rightarrow$ Emp(m, d)
### Chase Example

<table>
<thead>
<tr>
<th>ManagedBy</th>
<th>Emp</th>
<th>Dept</th>
</tr>
</thead>
<tbody>
<tr>
<td>emp</td>
<td>mgr</td>
<td></td>
</tr>
<tr>
<td>Barak</td>
<td>Ahuva</td>
<td></td>
</tr>
<tr>
<td>Chen</td>
<td>Ahuva</td>
<td></td>
</tr>
<tr>
<td>Ella</td>
<td>Doron</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- ManagedBy\((e, m)\) $\rightarrow$ Emp\((e, d)\), Dept\((d, m)\)
- Dept\((d, m)\), Dept\((d', m)\) $\rightarrow$ $d = d'$
- Dept\((d, m)\) $\rightarrow$ Emp\((m, d)\)
### Chase Example

<table>
<thead>
<tr>
<th>ManagedBy</th>
<th>Emp</th>
<th>Dept</th>
</tr>
</thead>
<tbody>
<tr>
<td>emp</td>
<td>mgr</td>
<td></td>
</tr>
<tr>
<td>Barak</td>
<td>Ahuva</td>
<td></td>
</tr>
<tr>
<td>Chen</td>
<td>Ahuva</td>
<td></td>
</tr>
<tr>
<td>Ella</td>
<td>Doron</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>emp</th>
<th>dept</th>
</tr>
</thead>
<tbody>
<tr>
<td>Barak</td>
<td>⊥₁</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>dept</th>
<th>mgr</th>
</tr>
</thead>
<tbody>
<tr>
<td>⊥₁</td>
<td>Ahuva</td>
</tr>
</tbody>
</table>

- ManagedBy\((e, m)\) → Emp\((e, d)\), Dept\((d, m)\)
- Dept\((d, m)\), Dept\((d', m)\) → \(d = d'\)
- Dept\((d, m)\) → Emp\((m, d)\)
### Chase Example

<table>
<thead>
<tr>
<th>ManagedBy</th>
<th>Emp</th>
<th>Dept</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>emp</strong></td>
<td><strong>emp</strong></td>
<td><strong>dept</strong></td>
</tr>
<tr>
<td>Barak</td>
<td></td>
<td>Barak</td>
</tr>
<tr>
<td>Chen</td>
<td></td>
<td>Chen</td>
</tr>
<tr>
<td>Ella</td>
<td></td>
<td>Ella</td>
</tr>
<tr>
<td><strong>mgr</strong></td>
<td></td>
<td><strong>mgr</strong></td>
</tr>
<tr>
<td>Ahuva</td>
<td></td>
<td>Ahuva</td>
</tr>
<tr>
<td>Ahuva</td>
<td></td>
<td>Ahuva</td>
</tr>
<tr>
<td>Doron</td>
<td></td>
<td>Doron</td>
</tr>
</tbody>
</table>

- \( \text{ManagedBy}(e, m) \rightarrow \text{Emp}(e, d), \text{Dept}(d, m) \)
- \( \text{Dept}(d, m), \text{Dept}(d', m) \rightarrow d = d' \)
- \( \text{Dept}(d, m) \rightarrow \text{Emp}(m, d) \)
### Chase Example

<table>
<thead>
<tr>
<th>ManagedBy</th>
<th>Emp</th>
<th>Dept</th>
</tr>
</thead>
<tbody>
<tr>
<td>emp</td>
<td>mgr</td>
<td>dept</td>
</tr>
<tr>
<td>Barak</td>
<td>Ahuva</td>
<td>(\perp_1)</td>
</tr>
<tr>
<td>Chen</td>
<td>Ahuva</td>
<td>(\perp_2)</td>
</tr>
<tr>
<td>Ella</td>
<td>Doron</td>
<td>(\perp_3)</td>
</tr>
</tbody>
</table>

- \(\text{ManagedBy}(e, m) \rightarrow \text{Emp}(e, d), \text{Dept}(d, m)\)
- \(\text{Dept}(d, m), \text{Dept}(d', m) \rightarrow d = d'\)
- \(\text{Dept}(d, m) \rightarrow \text{Emp}(m, d)\)
### Chase Example

<table>
<thead>
<tr>
<th>ManagedBy</th>
<th></th>
<th>Emp</th>
<th></th>
<th>Dept</th>
</tr>
</thead>
<tbody>
<tr>
<td>emp</td>
<td></td>
<td>emp</td>
<td></td>
<td>dept</td>
</tr>
<tr>
<td>mgr</td>
<td></td>
<td>emp</td>
<td></td>
<td>dept</td>
</tr>
<tr>
<td>Barak</td>
<td>Ahuva</td>
<td>Barak</td>
<td>⊥₁</td>
<td></td>
</tr>
<tr>
<td>Chen</td>
<td>Ahuva</td>
<td>Chen</td>
<td>⊥₂</td>
<td></td>
</tr>
<tr>
<td>Ella</td>
<td>Doron</td>
<td>Ella</td>
<td>⊥₃</td>
<td></td>
</tr>
</tbody>
</table>

- ManagedBy\((e, m) \rightarrow\) Emp\((e, d)\), Dept\((d, m)\)
- Dept\((d, m)\), Dept\((d', m)\) → \(d = d'\)
- Dept\((d, m)\) → Emp\((m, d)\)
## Chase Example

<table>
<thead>
<tr>
<th>ManagedBy</th>
<th>Emp</th>
<th>Dept</th>
</tr>
</thead>
<tbody>
<tr>
<td>emp</td>
<td>mgr</td>
<td>dept</td>
</tr>
<tr>
<td>Barak</td>
<td>Ahuva</td>
<td>( \bot_1 )</td>
</tr>
<tr>
<td>Chen</td>
<td>Ahuva</td>
<td>( \bot_2 )</td>
</tr>
<tr>
<td>Ella</td>
<td>Doron</td>
<td>( \bot_3 )</td>
</tr>
</tbody>
</table>

- ManagedBy(\( e, m \)) \( \rightarrow \) Emp(\( e, d \)) , Dept(\( d, m \))
- Dept(\( d, m \)) , Dept(\( d', m \)) \( \rightarrow d = d' \)
- Dept(\( d, m \)) \( \rightarrow \) Emp(\( m, d \))
### Chase Example

<table>
<thead>
<tr>
<th>ManagedBy</th>
<th></th>
<th></th>
<th>Dept</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>emp</td>
<td>mgr</td>
<td>dept</td>
</tr>
<tr>
<td></td>
<td>Barak</td>
<td>Ahuva</td>
<td>↓1</td>
</tr>
<tr>
<td></td>
<td>Chen</td>
<td>Ahuva</td>
<td>↓1</td>
</tr>
<tr>
<td></td>
<td>Ella</td>
<td>Doron</td>
<td>↓3</td>
</tr>
</tbody>
</table>

- $\text{ManagedBy}(e, m) \rightarrow \text{Emp}(e, d), \text{Dept}(d, m)$
- $\text{Dept}(d, m), \text{Dept}(d', m) \rightarrow d = d'$
- $\text{Dept}(d, m) \rightarrow \text{Emp}(m, d)$
### Chase Example

<table>
<thead>
<tr>
<th>ManagedBy</th>
<th>Emp</th>
<th>Dept</th>
</tr>
</thead>
<tbody>
<tr>
<td>emp</td>
<td>mgr</td>
<td></td>
</tr>
<tr>
<td>Barak</td>
<td>Ahuva</td>
<td></td>
</tr>
<tr>
<td>Chen</td>
<td>Ahuva</td>
<td></td>
</tr>
<tr>
<td>Ella</td>
<td>Doron</td>
<td></td>
</tr>
<tr>
<td>emp</td>
<td>dept</td>
<td></td>
</tr>
<tr>
<td>Barak</td>
<td>$\bot_1$</td>
<td></td>
</tr>
<tr>
<td>Chen</td>
<td>$\bot_1$</td>
<td></td>
</tr>
<tr>
<td>Ella</td>
<td>$\bot_3$</td>
<td></td>
</tr>
<tr>
<td>dept</td>
<td>mgr</td>
<td></td>
</tr>
<tr>
<td>$\bot_1$</td>
<td>Ahuva</td>
<td></td>
</tr>
<tr>
<td>$\bot_3$</td>
<td>Doron</td>
<td></td>
</tr>
</tbody>
</table>

- $\text{ManagedBy} (e, m) \rightarrow \text{Emp} (e, d), \text{Dept} (d, m)$
- $\text{Dept} (d, m), \text{Dept} (d', m) \rightarrow d = d'$
- $\text{Dept} (d, m) \rightarrow \text{Emp} (m, d)$
Chase Example

<table>
<thead>
<tr>
<th>ManagedBy</th>
<th>Emp</th>
<th>Dept</th>
</tr>
</thead>
<tbody>
<tr>
<td>emp</td>
<td>emp</td>
<td>dept</td>
</tr>
<tr>
<td></td>
<td>mgr</td>
<td></td>
</tr>
<tr>
<td>Barak</td>
<td></td>
<td>↓1</td>
</tr>
<tr>
<td>Chen</td>
<td></td>
<td>↓1</td>
</tr>
<tr>
<td>Ella</td>
<td></td>
<td>↓3</td>
</tr>
<tr>
<td>Ahuva</td>
<td></td>
<td>↓1</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- ManagedBy\((e, m)\) → Emp\((e, d)\), Dept\((d, m)\)
- Dept\((d, m)\), Dept\((d', m)\) → \(d = d'\)
- Dept\((d, m)\) → Emp\((m, d)\)
Chase Example

- ManagedBy\((e, m)\) → Emp\((e, d)\), Dept\((d, m)\)
- Dept\((d, m)\), Dept\((d', m)\) → \(d = d'\)
- Dept\((d, m)\) → Emp\((m, d)\)
Chase Example

<table>
<thead>
<tr>
<th>ManagedBy</th>
<th>Emp</th>
<th>Dept</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>emp</td>
<td></td>
<td></td>
</tr>
<tr>
<td>mgr</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Barak</td>
<td>emp</td>
<td>dept</td>
</tr>
<tr>
<td>Chen</td>
<td>emp</td>
<td>dept</td>
</tr>
<tr>
<td>Ella</td>
<td>emp</td>
<td>dept</td>
</tr>
</tbody>
</table>

- ManagedBy(e, m) → Emp(e, d), Dept(d, m)
- Dept(d, m), Dept(d', m) → d = d'
- Dept(d, m) → Emp(m, d)
Chase Failure and Termination

There are three kinds of chase executions:

1. Termination with *success*
2. Termination with *failure*
3. Non-termination
Chase Failure and Termination

There are three kinds of chase executions:

1. Termination with *success*
2. Termination with *failure*
3. Non-termination

   ▶ *How can that be?*
Chase Failure and Termination

There are three kinds of chase executions:

1. Termination with *success*
2. Termination with *failure*
3. Non-termination

   - *How can that be?*
   - $\text{Person}(x) \rightarrow \text{ChildOf}(x, y) \quad \text{ChildOf}(x, y) \rightarrow \text{ChildOf}(y, z)$
**Theorem [FKMP05]**

Let $\mathcal{M} = (S, T, \Sigma)$ be a schema mapping and $I$ a source instance.
**Theorem [FKMP05]**

Let $\mathcal{M} = (S, T, \Sigma)$ be a schema mapping and $I$ a source instance.

- If the chase terminates successfully, then the result is a universal solution.
Chase Gives a Universal Solution

**Theorem [FKMP05]**

Let $\mathcal{M} = (S, T, \Sigma)$ be a schema mapping and $I$ a source instance.

- If the chase terminates successfully, then the result is a universal solution.
- If the chase terminates with failure, then there is no solution at all.
Theorem [FKMP05]

Let $\mathcal{M} = (S, T, \Sigma)$ be a schema mapping and $I$ a source instance.

- If the chase terminates successfully, then the result is a universal solution.
- If the chase terminates with failure, then there is no solution at all.

The result of a successful chase is called a *canonical* universal solution.
We would like to be able to test whether a schema mapping \((S, T, \Sigma)\) is such that the chase is guaranteed to terminate
We would like to be able to test whether a schema mapping \((S, T, \Sigma)\) is such that the chase is guaranteed to terminate. Unfortunately, this is provably impossible.
The following problems are undecidable. Given a schema mapping \((S, T, \Sigma)\) and a source instance \(I\):

1. Is there any solution?
2. Is there any universal solution?
3. Is there any terminating chase?
4. Is every chase terminating?
Weak acyclicity is a property of a schema mapping that guarantees termination of the chase (and further consequent goodness properties).
Let \((S, T, \Sigma)\) be a schema mapping

- A *position* is an expression of the form \(R.A\), where \(R\) is a relation name and \(A\) is an attribute of \(R\)
Let \((S, T, \Sigma)\) be a schema mapping

- A \textit{position} is an expression of the form \(R.A\), where \(R\) is a relation name and \(A\) is an attribute of \(R\)

- The \textit{dependency graph} of \(\Sigma\) is the directed graph \(G_{\Sigma}\) defined as follows:
- Let \((S, T, \Sigma)\) be a schema mapping
- A *position* is an expression of the form \(R.A\), where \(R\) is a relation name and \(A\) is an attribute of \(R\)
- The *dependency graph* of \(\Sigma\) is the directed graph \(G_{\Sigma}\) defined as follows:
  - The nodes are the positions
Let \((S, T, \Sigma)\) be a schema mapping

- A **position** is an expression of the form \(R.A\), where \(R\) is a relation name and \(A\) is an attribute of \(R\)

- The **dependency graph** of \(\Sigma\) is the directed graph \(G_\Sigma\) defined as follows:
  - The nodes are the positions
  - For each t-TGD \(\varphi(x) \rightarrow \exists y \psi(x, y)\), position \(p\) at \(\varphi(x)\), position \(q\) at \(\psi(x, y)\) there is an edge \(p \rightarrow q\) if:
    - \(p\) and \(q\) have the same variable (ordinary edge)
    - \(q\) has an existential variable (special edge)
Example

- ManagedBy(e, m) \rightarrow Emp(e, d), Dept(d, m)
- Dept(d, m) \rightarrow Emp(m, d)
- Emp(e, d) \rightarrow Dept(d, m)

The chase:

```
Dept.dept  \leftrightarrow  Emp.dept
Dept.mgr  \rightarrow  Emp.emp
```
Weak Acyclicity

- $\Sigma$ is \textit{weakly acyclic} if $G_\Sigma$ does not have any cycle that goes through a special edge.
Weak Acyclicity

- $\Sigma$ is *weakly acyclic* if $G_\Sigma$ does not have any cycle that goes through a special edge
- Note: weak acyclicity is a property of the t-TGDs (independent of the st-TGDs and t-EGDs)
Weak Acyclicity

- $\Sigma$ is **weakly acyclic** if $G_\Sigma$ does not have any cycle that goes through a special edge
- Note: weak acyclicity is a property of the t-TGDs (independent of the st-TGDs and t-EGDs)
- Weak acyclicity can be tested in polynomial time in $\Sigma$

*Why?*
Example: Weakly Acyclic

- ManagedBy\((e, m) \rightarrow \text{Emp}(e, d), \text{Dept}(d, m)\)
- Dept\((d, m) \rightarrow \text{Emp}(m, d)\)

Dept.dept $\rightarrow$ Emp.dept

Dept.mgr $\rightarrow$ Emp.emp
Example: Not Weakly Acyclic

- ManagedBy\((e, m) \rightarrow \text{Emp}(e, d), \text{Dept}(d, m)\)
- Dept\((d, m) \rightarrow \text{Emp}(m, d)\)
- Emp\((e, d) \rightarrow \text{Dept}(d, m)\)

```
Dept.dept ←→ Emp.dept
Dept.mgr  → Emp.emp
```
Weak Acyclicity and Chase Termination

**Theorem [FKMP05]**

Let $\mathcal{M} = (S, T, \Sigma)$ be a schema mapping. Assume that $\Sigma$ is weakly acyclic. Given a source instance $I$, the chase terminates in polynomial time in $|I|$. 
**Theorem [FKMP05]**

Let $\mathcal{M} = (S, T, \Sigma)$ be a schema mapping. Assume that $\Sigma$ is weakly acyclic. Given a source instance $I$, the chase terminates in polynomial time in $|I|$. (The degree of the polynomial is determined by $\mathcal{M}$.)
**Theorem [FKMP05]**

Let $\mathcal{M} = (S, T, \Sigma)$ be a schema mapping and $I$ a source instance. Assume that $\Sigma$ is weakly acyclic. Then the following are equivalent:

1. There is a solution.
2. There is a universal solution.
3. The chase terminates successfully.
Theorem [FKMP05]

Let $\mathcal{M} = (S, T, \Sigma)$ be a schema mapping and $I$ a source instance. Assume that $\Sigma$ is weakly acyclic. Then the following are equivalent:

1. There is a solution.
Theorem [FKMP05]

Let $\mathcal{M} = (S, T, \Sigma)$ be a schema mapping and $I$ a source instance. Assume that $\Sigma$ is weakly acyclic. Then the following are equivalent:

1. There is a solution.
2. There is a universal solution.
Theorem [FKMP05]

Let $\mathcal{M} = (S, T, \Sigma)$ be a schema mapping and $I$ a source instance. Assume that $\Sigma$ is weakly acyclic. Then the following are equivalent:

1. There is a solution.
2. There is a universal solution.
3. The chase terminates successfully.
We now have an algorithm for computing the certain answers, given a source instance \( I \) for a schema mapping \((S, T, \Sigma)\):

```plaintext
chase \( I \) and \( \Sigma \)
if the chase fails
then abort("no solution")
else
let \( J \) be the resulting solution
return \( Q(J) \)
```

Of course, the algorithm assumes that the chase terminates. If \( \Sigma \) is weakly acyclic, we get a polynomial-time algorithm.
We now have an algorithm for computing the certain answers, given a source instance $I$ for a schema mapping $(S, T, \Sigma)$:

```
chase $I$ and $\Sigma$
if the chase fails then
  abort("no solution")
else
  let $J$ be the resulting solution
  return $Q(J)$
```

Of course, the algorithm assumes that the chase terminates. If $\Sigma$ is weakly acyclic, we get a polynomial-time algorithm.
We now have an algorithm for computing the certain answers, given a source instance $I$ for a schema mapping $(S, T, \Sigma)$:

\begin{algorithm}
chase $I$ and $\Sigma$
if the chase fails then
abort(“no solution”) 
else 
let $J$ be the resulting solution
return $Q(J)$
\end{algorithm}

Of course, the algorithm assumes that the chase terminates.
We now have an algorithm for computing the certain answers, given a source instance $I$ for a schema mapping $(S, T, \Sigma)$:

\[
\text{chase } I \text{ and } \Sigma \\
\text{if the chase fails then} \\
\text{\quad abort(“no solution”)} \\
\text{else} \\
\text{\quad let } J \text{ be the resulting solution} \\
\text{\quad return } Q(J) \\
\]

- Of course, the algorithm assumes that the chase terminates
- If $\Sigma$ is weakly acyclic, we get a polynomial-time algorithm
Can we construct a core solution in polynomial time if $\Sigma$ is weakly acyclic?
Can we construct a core solution in polynomial time if $\Sigma$ is weakly acyclic?

- The straightforward way is:
  1. Compute a universal solution $J$ via chase
  2. Compute the core of $J$
Computing a Core Solution

- Can we construct a core solution in polynomial time if $\Sigma$ is weakly acyclic?
- The straightforward way is:
  1. Compute a universal solution $J$ via chase
  2. Compute the core of $J$
- Does it return the correct result?
Can we construct a core solution in polynomial time if $\Sigma$ is weakly acyclic?

The straightforward way is:
1. Compute a universal solution $J$ via chase
2. Compute the core of $J$

Does it return the correct result?

Is it a polynomial-time strategy?
Can we construct a core solution in polynomial time if $\Sigma$ is weakly acyclic?

The straightforward way is:
1. Compute a universal solution $J$ via chase
2. Compute the core of $J$

Does it return the correct result?

Is it a polynomial-time strategy?

What is the complexity of finding the core of a $v$-instance?
We show that it is NP-hard to compute the core of a $v$-instance.
Reduction from 3-Colorability (1)

- We show that it is NP-hard to compute the core of a $v$-instance
- Reduction from 3-colorability: given a graph $g$, is there a legal node coloring by 3 colors?
Reduction from 3-Colorability (1)

- We show that it is NP-hard to compute the core of a \( v \)-instance.
- Reduction from 3-colorability: given a graph \( g \), is there a legal node coloring by 3 colors?
- Denote \( g = (V, E) \) where \( V = \{v_1, \ldots, v_n\} \).
Reduction from 3-Colorability (1)

- We show that it is NP-hard to compute the core of a \( v \)-instance
- Reduction from 3-colorability: given a graph \( g \), is there a legal node coloring by 3 colors?
- Denote \( g = (V, E) \) where \( V = \{v_1, \ldots, v_n\} \)
- Construct a \( v \)-instance \( D \) with a single relation \( R_E \) that consists of only labeled nulls
  - \( R(\bot_i, \bot_j) \) for all \( i < j \) such that \( \{v_i, v_j\} \) is an edge
  - \( R(\bot_x, \bot_y) \) for every \( x \neq y \) in \( \{r, g, b\} \)
    - Denote this part of \( D \) as \( D_\Delta \)
Claim: the core of $D$ is $D_\Delta$ if and only if $g$ is 3-colorable
Reduction from 3-Colorability (2)

- Claim: the core of $D$ is $D_\Delta$ if and only if $g$ is 3-colorable
- Proof steps:
Reduction from 3-Colorability (2)

- Claim: the core of $D$ is $D_{\Delta}$ if and only if $g$ is 3-colorable
- Proof steps:
  - If $\text{Core}(D) = D_{\Delta}$ then the endomorphism is a legal 3-coloring
Claim: the core of $D$ is $D_\Delta$ if and only if $g$ is 3-colorable

Proof steps:
- If $\text{Core}(D) = D_\Delta$ then the endomorphism is a legal 3-coloring
- A 3-coloring defines an endomorphism to $D_\Delta$
Claim: the core of $D$ is $D_\Delta$ if and only if $g$ is 3-colorable

Proof steps:
- If $\text{Core}(D) = D_\Delta$ then the endomorphism is a legal 3-coloring
- A 3-coloring defines an endomorphism to $D_\Delta$
- $\text{Core}(D_\Delta) = D_\Delta$
Reduction from 3-Colorability (2)

- Claim: the core of $D$ is $D_\Delta$ if and only if $g$ is 3-colorable
- Proof steps:
  - If $\text{Core}(D) = D_\Delta$ then the endomorphism is a legal 3-coloring
  - A 3-coloring defines an endomorphism to $D_\Delta$
  - $\text{Core}(D_\Delta) = D_\Delta$
- This establishes NP-hardness
Claim: the core of $D$ is $D_\Delta$ if and only if $g$ is 3-colorable

Proof steps:
- If $\text{Core}(D) = D_\Delta$ then the endomorphism is a legal 3-coloring
- A 3-coloring defines an endomorphism to $D_\Delta$
- $\text{Core}(D_\Delta) = D_\Delta$

This establishes NP-hardness

Nevertheless, $D$ is not necessarily the result of chasing with a weakly acyclic $\Sigma$!
Reduction from 3-Colorability (2)

- Claim: the core of $D$ is $D_\Delta$ if and only if $g$ is 3-colorable
- Proof steps:
  - If $\text{Core}(D) = D_\Delta$ then the endomorphism is a legal 3-coloring
  - A 3-coloring defines an endomorphism to $D_\Delta$
  - $\text{Core}(D_\Delta) = D_\Delta$
- This establishes NP-hardness
- Nevertheless, $D$ is not necessarily the result of chasing with a weakly acyclic $\Sigma$!
- Gottlob and Nash [GN08] proved that, if $J$ is the result of such a chase, then $\text{Core}(D)$ can, in fact, be computed in polynomial time.
Finding Core Solutions

**Theorem [GN08]**

Let $\mathcal{M} = (S, T, \Sigma)$ be a schema mapping. Assume that $\Sigma$ is weakly acyclic. Given a source instance $I$, a core solution can be found in polynomial time, if any solution exists.


