Universität Rostock

Traditio et Innovatio

The Theory behind Minimizing Research Data

Problem

Application

- Different applications dealing with growing amounts of data:
  - Research data management with measurement data
  - Sensor data management for smart (assistive) systems aiming at the derivation of activity and intention models by means of Machine Learning algorithms
- Aim: Describing traceability, reconstructibility and replicability of the path from data collection to publication

Aim of our research project

- Reducing the primary measurement or sensor data to an important kernel
- Calculating the kernel even after updating databases or database schemes

$\Rightarrow$ Minimizing the sub-database that has to be stored to guarantee the reproducibility of the performed evaluation

Unification of Provenance and Evolution

- Goal: Performing provenance queries Q_{prov} after evolution $\mathcal{E}$ of databases and database schemes
- Idea: Combination of provenance with schema and data evolution
- Wanted: New minimal sub-database to be archived $J^* \subseteq J$
  $\Rightarrow$ Calculation of a new query $Q'(J(S_3))$ from the old query $Q(I(S_1))$

Example

- Schemas: S_1, S_2 and S_3
- Query: Q with minimal sub-database $I^* \subseteq I$
- Provenance Query: Q_{prov} with input $K^* \subseteq K$
- Schema evolution: $\mathcal{E}$ with minimal sub-database $J^* \subseteq J$

Calculation of a minimal part of the database (minimal sub-database)

- Different constraints for the sub-database to be determined:
  - Number of tuples of the original relation remains unchanged.
  - The sub-database can be mapped homomorphically to the original database.
  - The sub-database is an intensional description of the original database.
- Question: Which additional information is required to be able to reconstruct the minimal part I^* of the database I if the result and the evaluation query Q are both archived?
- Idea: Calculation of an inverse query Q_{prov} with input $K^* \subseteq K$ to determine the minimal sub-database
  $\Rightarrow$ Type of inverse depending on the additional information noted

Example

- Schemas: S_1, S_2 and S_3
- Query: $Q = \textbf{AVG}(\texttt{grade})$
- Minimal sub-databases:
  - $I_a^*(S_1) \subseteq I(S_1)$ without extension $K'(S_2')$
  - $I_b^*(S_1) = I(S_1)$ with extension $K'(S_2')$
- Provenance Query: $Q_{\text{prov}} = \textbf{AVG}^{-1}(\texttt{grade})$
- Input for Q_{prov}: $K^*(S_2) = K(S_2)$
  $\Rightarrow$ existence of a
  - result equivalent CHASE-inverse for I_a^*
  - tp-relaxed CHASE-inverse for I_b^*
  - exact CHASE-inverse for I_c^*

$I_a^*(S_1)$:

id	module	grade	
η_{id_1}	η_{module_1}	1.725	t_1

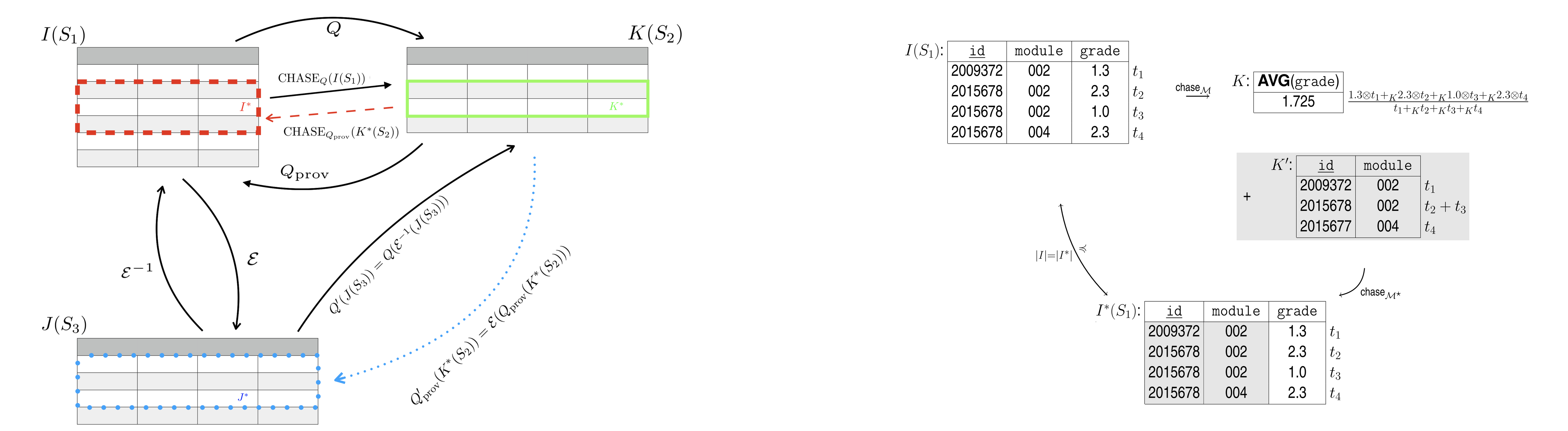
$I_b^*(S_1)$:

id	module	grade	
η_{id_1}	η_{module_1}	1.3	t_1
η_{id_2}	η_{module_2}	2.3	t_2
η_{id_3}	η_{module_3}	1.0	t_3
η_{id_4}	η_{module_4}	2.3	t_4

$I_c^*(S_1)$:

id	module	grade	
2009372	002	1.3	t_1
2015678	002	2.3	t_2
2015678	002	1.0	t_3
2015678	004	2.3	t_4

Provenance Management and Schema Evolution:



CHASE-inverse schema mappings

Combining the techniques

- CHASE:
  - CHASE incorporates dependencies $\star$ in an object $\bigcirc$, i.e.
    $$\text{chase}_\star(\bigcirc) = \circledast$$
  - Source-to-target tuple-generating dependency (s-t tgd):
    $$\forall \mathbf{x} : (\phi(\mathbf{x}) \rightarrow \exists \mathbf{y} : \psi(\mathbf{x}, \mathbf{y}))$$
    $\Rightarrow$ Express the evaluation query Q as a schema mapping $\mathcal{M} = (S_1, S_2, \Sigma)$ with source and target schemas S_1 and S_2 and a set of dependencies Σ
- Provenance Management: traceability of a result back to the relevant original data
- CHASE&BACKCHASE:

Schema:	$S_1 \frown S_2 \frown S_1$
Instance:	$I \xrightarrow{\text{chase}_{\mathcal{M}}} K \xrightarrow{\text{chase}_{\mathcal{M}^*}} I^*$

Types of CHASE-Inverses

- CHASE-types:
  - Exact CHASE-inverse: Reconstructs the complete original database
  - Tuple preserving relaxed CHASE-inverse: Preserves the number of tuples
  - Result equivalent CHASE-inverse: $\text{chase}_{\mathcal{M}}(I) = \text{chase}_{\mathcal{M}}(I^*)$
- Reduction: result equivalent $\preceq$ relaxed $\preceq$ tp-relaxed $\preceq$ exact
- Conditions for the existence of CHASE inverse:

CHASE inverse	sufficient condition	necessary condition
Exact	-	$I^* = I$
Classical	Exact CHASE-inverse	$I^* \equiv I$
Tp-relaxed	Exact CHASE-inverse	$I^* \preceq I, \mid I^* \mid = \mid I \mid$
Relaxed	Tp-relaxed CHASE inverse	$I^* \preceq I$
Result equivalent	Relaxed CHASE-inverse	$I^* \leftrightarrow_{\mathcal{M}} I$

Tanja Auge , Andreas Heuer

INSTITUTE OF COMPUTER SCIENCE I DATABASE RESEARCH GROUP I

University of Rostock, Albert-Einstein-Straße 22 I 18059 Rostock, Germany