Humboldt-Universität Informatik

Humboldt Universit t zu Berlin
Institut f r Informatik


# Query Translation from XPath to SQL in the Presence of Recursive DTDs

VL XML, XPath, XQuery: Neue Konzepte für Datenbanken

Jörg Pohle, pohle@informatik.hu-berlin.de
Daniel Apelt, apelt@informatik.hu-berlin.de

# Überblick

- Das Problem
- Unser Beispiel
- Ablauf des Algorithmus
- XPath → Reg. XPath
- Zurück zum Beispiel
- Simple LFP Operator
- Reg. XPath → SQL
- Zurück zum Beispiel

# Das Problem

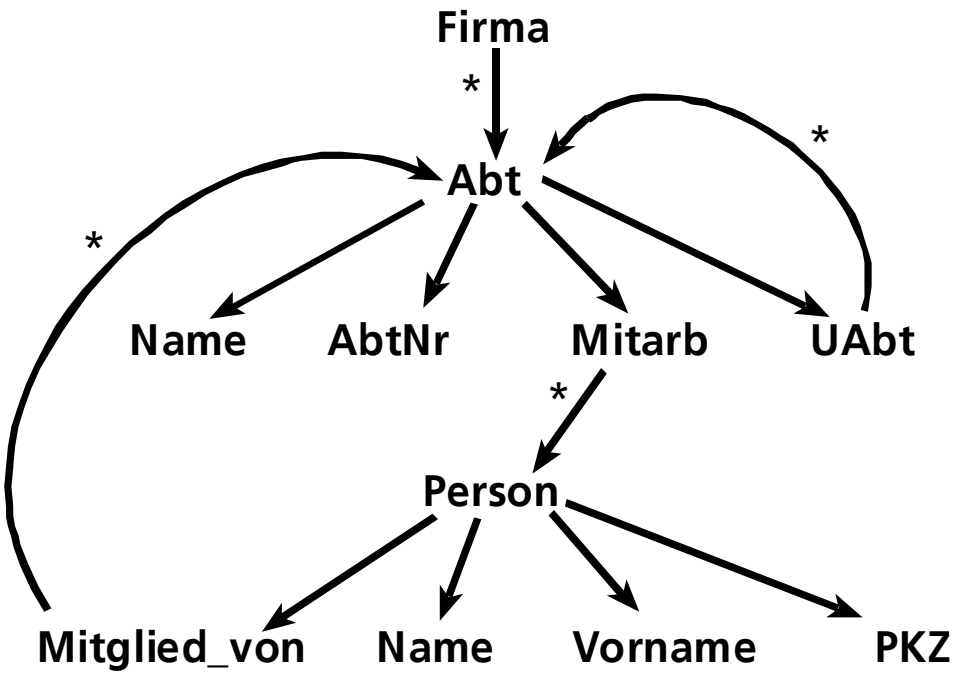
## Eine formale Beschreibung

Gegeben: DTD $D$
Mapping $\tau_d: D \rightarrow R$
Relationenschema $R = \tau(D)$
Xpath Query $Q$

Gesucht: SQL Query $Q'$,
so daß für jedes XML-Dokument $T$
mit $T$ valid zu $D$ gilt:

$$Q(T) = Q'(\tau(T))$$

# Unser Beispiel




$R_{Firma}$ (From, $ID_{Firma}$)

$R_{Abt}$ (From, $ID_{Abt}$, AbtNr, Name, Mitarb, UAbt)

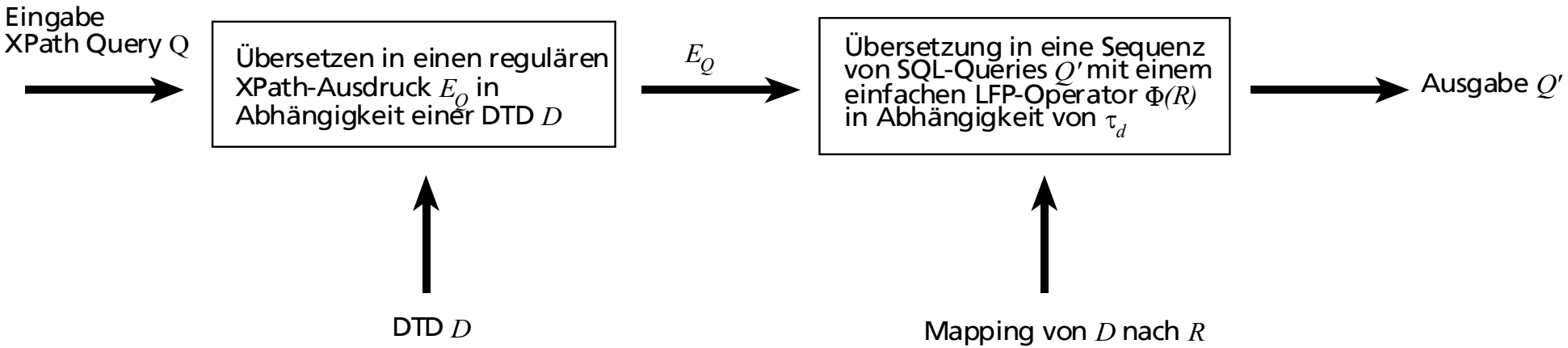
$R_{Person}$ (From, $Id_{Person}$, Mitglied_von, Name, Vorname, PKZ)

# Beispiel-Query

XPath Query $Q$ = `//Abt/Name`

SQL Query $Q'$ = ?

# Ablauf des Algorithmus




Grafik nach: Query Translation from XPath to SQL in the Presence of Recursive DTDs

# XPath → Reg. XPath 1/2

**Ziel:** Übersetzung einer XPath-Query $Q$ in eine reguläre XPath-Query $E_Q$, wobei $E_Q(T) = Q(T)$

**Ablauf:**

- lokale Übersetzung für jede SubQuery
  $E_p$ = x2r($p$,$A$)
- Zusammenfügen der lokalen Übersetzungen
  $E_Q$ = x2r($Q$,$r$), wobei $r$ Wurzel von $D$ ist

**Herausforderung:** Umgang mit Zyklen in $D$

# XPath → Reg. XPath 2/2

## Umgang mit Zyklen in $D$

**1. Tarjan:** teilen des Graphen in Zusammenhangs-komponenten, Berechnen der regulären Ausdrücke mit Hilfe der Gaußschen Elimination und Zusammenführung der Lösungen

**2. Cycle Contraction:**

- suchen von Kreisen $C_j$, danach suchen von A-B-Pfaden $L_1,...,L_n$
- Kodieren von $L_i = A \rightarrow ... \rightarrow B$ in $E_i = A/.../B$
- für jeden Kreis $C_j$ verbunden mit Knoten $A_i$, $A_i$ wird ersetzt durch $A_i/E^*_{Cj}$

# Zurück zum Beispiel

$C_1$: Abt → Mitarb → Person → Mitglied_von → Abt
$C_2$: Abt → UAbt → Abt

A-B-Pfad //Abt: Firma → Abt

$E_1$ = Firma/Abt
= Firma/Abt/$E^*_C$
$E_2$ = Name

$E_Q$ = Firma/Abt/$E^*_C$/Name mit $E^*_C = (E_{C1} \cup E_{C2})^*$

# Simple LFP Operator 1/2

Problem: Kleenesche Hülle in SQL?

Entweder:
*with...recursive* von SQL99
(nur in IBM DB2 unterstützt)

Oder:
mit Simple LFP Operator
(IBM DB2, Oracle, Microsoft)

# Simple LFP Operator 2/2

Fixpunkt: Punkt, der bei einer Abbildung mit sich selbst zusammenfällt $(f(x)=x)$

Simple LFP Operator: $\Phi(R)$

$$R^0 \leftarrow R$$

$$R^i \leftarrow R^{i-1} \cup (R^{i-1}\ join_C\ R^0)$$

# Reg. XPath → SQL

- findet alle Teilausdrücke von $E_Q$ und sortiert sie topologisch in aufsteigender Ordnung
- mit dem innersten Ausdruck beginnend wird dann die SQL-Query zusammen gebaut (versch. Fälle)
- Einschränken auf das was von $r$ erreicht werden kann
- Optimieren des Ergebnis durch Eliminieren von leeren Mengen und Extrahieren von common subqueries

# Zurück zum Beispiel

$$R_{C_1} \leftarrow \Pi_{R_P.F,\, R_A.ID}\left(R_P \; join_{R_P.ID = R_A.F} \; R_A\right)$$

$$R_{C_2} \leftarrow R_A$$

$$R \leftarrow R_{C_1} \cup R_{C_2}$$

$$R_\gamma \leftarrow \Phi(R) \cup \Pi_{ID,\, ID}(R_C)$$

$$R_E \leftarrow \Pi_{R_{A.Name}}\left(R_F \; join_{R_{P.ID} = R_{A.F}} \; R_A \; join_{R_{A.ID} = R_{\gamma}.F} \; R_\gamma \; join_{R_\gamma.ID = R_{A.F}} \; R_{A.ID}\right)$$

# Zusammenfassung

Über einer (rekursiven) DTD $D$ wird aus einer (rekursiven) XPath-Query $Q$ über den Zwischenschritt einer regulären XPath-Query $E_Q$ eine SQL-Abfrage $Q'$ erzeugt.

Es wird dabei eine SQL-Abfrage erzeugt, die von (fast) allen kommerziellen RDBMS unterstützt wird.

# Literatur


Wenfei Fany, Jeffrey Xu Yu, Hongjun Lu, Jianhua Lu, Rajeev Rastogi: *Query Translation from XPath to SQL in the Presence of Recursive DTDs* in VLDB, 2005 (& Slides)

Rajasekar Krishnamurthy, Venkatesan T. Chakaravarthy, Raghav Kaushik, Jeffrey F. Naughton: *Recursive XML Schemas, Recursive XML Queries, and Relational Storage: XML-to-SQL Query Translation*; University of Wisconsin-Madison; 2004;
http://www.cs.wisc.edu/~sekar/research/recursiveqt.pdf

Robert Endre Tarjan: *Fast Algorithms for Solving Path Problems* JACM, Volume 28, Issue 3, p. 594 - 614; 1981

Stephan Kreutzer, Nicole Schweikardt: *Logik und Informatik*; it - Information Technology 46 (2004) 3;
http://www2.informatik.hu-berlin.de/lds/publications/KreutzerSchweikardt_it2004.pdf


# Vielen Dank für Eure Aufmerksamkeit

# „SQLGen-R“

- $A_S$ – FA für XML-Schema
- $A_Q$ – FA für XPath-Query
- $A_S \times A_Q = A_{SQ}$ – FA

- Partitionierung von $A_{SQ}$ in starke Zusammenhangskomponenten $c_1,\ldots,c_n$ und $c_o$ (linearer Rest)
- für die Partitionen werden SQL-Abfragen berechnet und dann zusammengefügt