Non-Standard-Datenbanken

Probabilistische Datenbanken

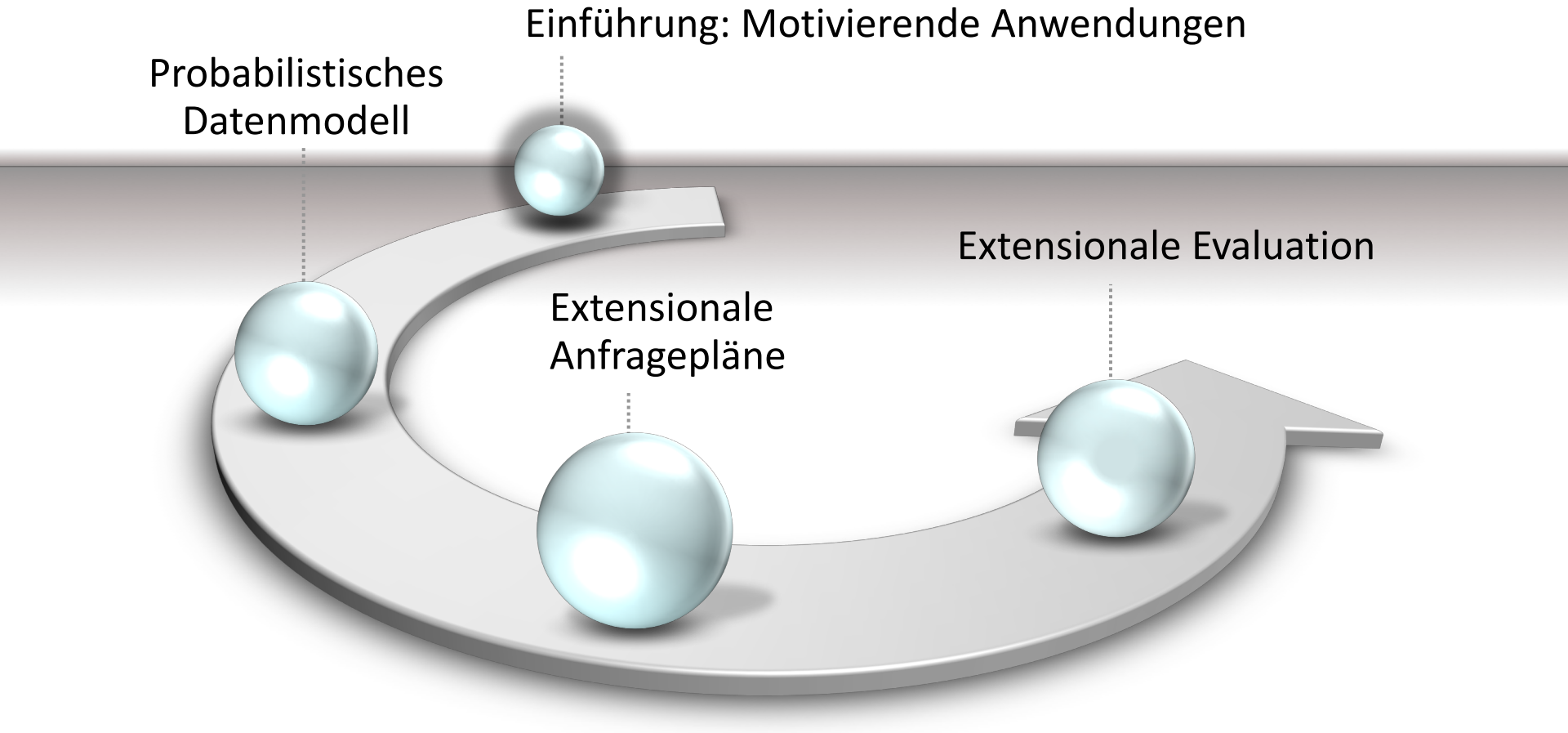
Prof. Dr. Ralf Möller

Universität zu Lübeck

Institut für Informationssysteme

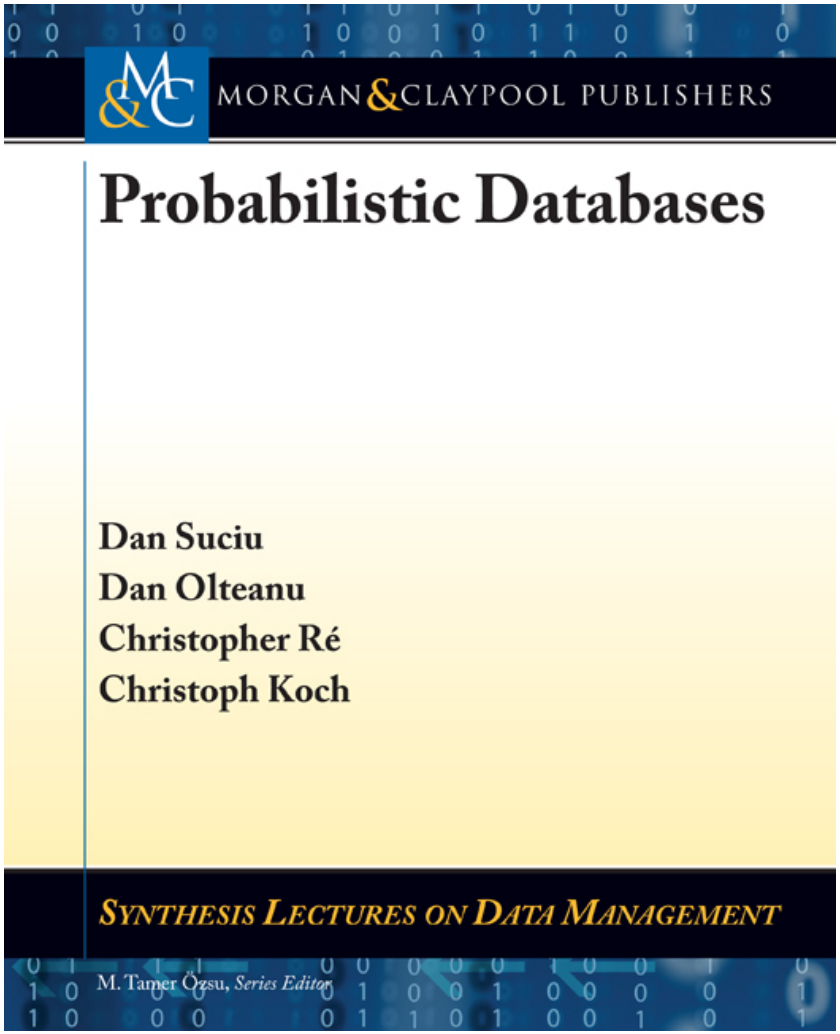
Non-Standard-Datenbanken

Probabilistische Datenbanken



Danksagung

Die Präsentationen sind nach einem Tutorial von Dan Suciu gestaltet und basieren auf dem Lehrbuch Probabilistic Databases



Probabilistische Datenbanken

- Daten: Relationale Daten plus Wahrscheinlichkeiten, um Grad der Unsicherheit auszudrücken
- Anfragen: SQL-Anfragen, deren Antworten annotiert sind mit Ausgabewahrscheinlichkeiten
- Ermöglicht Ihnen einen neuen Blick auf beides, Datenbanken und Wahrscheinlichkeiten

Beispiel 1: Informationsextraktion

52-A Goregaon West Mumbai 400 076

Extraktion

Standard DB: Speichere nur wahrscheinlichste Extraktion

Id	House_no	Area	City	Pincode	Prob
1	52	Goregaon West	Mumbai	400 062	0.1
1	52-A	Goregaon	West Mumbai	400 062	0.2
1	52-A	Goregaon West	Mumbai	400 062	0.5
1	52	Goregaon	West Mumbai	400 062	0.2

Probabilistische DB: Speichert die meisten/alle Extraktionen um Recall zu erhöhen

Kernidee: Wahrscheinlichkeiten gegeben durch Extraktion korrelieren mit der Präzision der Extraktion

Gupta, Sarawagi: Creating Probabilistic Databases from Information Extraction Models. VLDB **2006**

Beispiel 2: Modellierung fehlender Daten

id	age	edu	inc	nw
t_1	20	HS	?	?
t_2	20	BS	50K	100K
t_3	20	?	50K	?
t_4	20	HS	100K	500K
t_5	20	?	?	?
t_6	20	HS	50K	100K
t_7	20	HS	50K	500K
t_8	?	HS	?	?
t_9	30	BS	100K	100K
t_{10}	30	?	100K	?
t_{11}	30	HS	?	?
t_{12}	30	MS	?	?
t_{13}	40	BS	100K	100K
t_{14}	40	HS	?	?
t_{15}	40	BS	50K	500K
t_{16}	40	HS	?	500K
t_{17}	40	HS	100K	500K

Standard-DB: NULL

Probabilistische DB:
Verteilung auf mögl. Werten

id	age	edu	inc	nw	prob
$t_{12}.1$	30	MS	50K	100K	0.30
$t_{12}.2$	30	MS	50K	500K	0.45
$t_{12}.3$	30	MS	100K	100K	0.10
$t_{12}.4$	30	MS	100K	500K	0.15

Kernidee:
Inferiere Verteilung für fehlende Daten.

Stoyanovich, Davidson, Milo, Tannen: Deriving probabilistic databases with inference ensembles. ICDE **2011**

Beispiel 3: Datenreinigung

Unclean Data

ID	Name	ZIP	Birth Date
1	Green	51359	781310
2	Green	51358	781210
3	Peter	30128	870932
4	Peter	30128	870932
5	Gree	51359	19771210
6	Chuck	51359	19460924

Single Repair

{1,2}
{3,4}
{5}
{6}

One-shot Duplicate Detection

Uncertain Duplicate Detection

Multiple Repairs

{1,2,5}	{1,2}	{1,5,6}
{3,4}	{3,4}	{2}
{6}	{5}	{3,4}
	{6}	

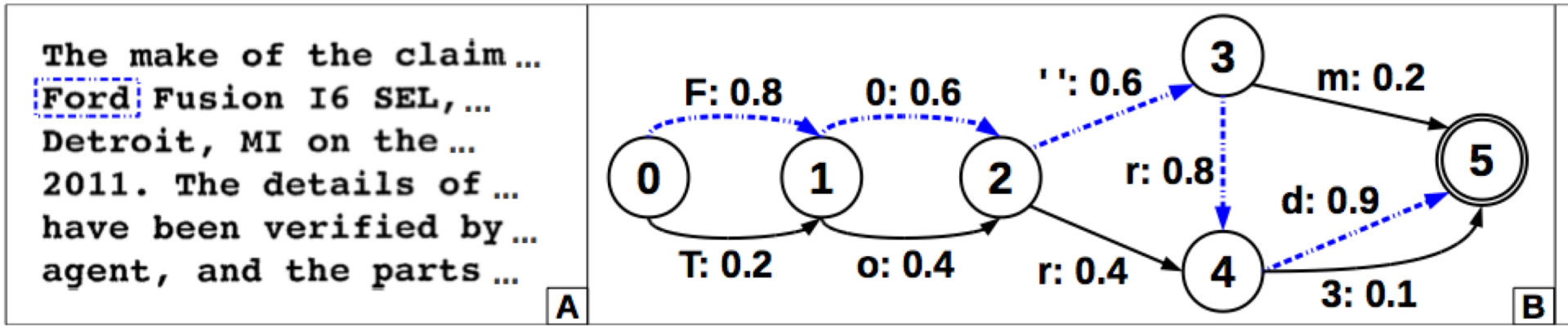
Standard-DB:
Reinigung bedeutet, eine mögliche Reparatur zu wähen

Probabilistische DB
Speichere viele/alle möglichen Reparaturen

Herausforderung: Representation von multiplen Reparaturen

Beskales, Soliman, Ilyas, Ben-David: Modeling and Querying Possible Repairs in Duplicate Detection. PVLDB **2009**

Beispiel 4: OCR



Verwendung von OCRopus von Google Books: Ausgabe ist stochastischer Automat
Üblicherweise wird nur Maximum Apriori Estimate (MAP) gespeichert
Mit probabilistischer Databasis: Speicherung verschiedener Möglichkeiten: Erhöhe Recall.

```
SELECT DocId, Loss
FROM Claims
WHERE Year = 2010
    AND DocData LIKE '%Ford%';
```

Kumar, Ré: Probabilistic Management of OCR Data using an RDBMS. PVLDB **2011**

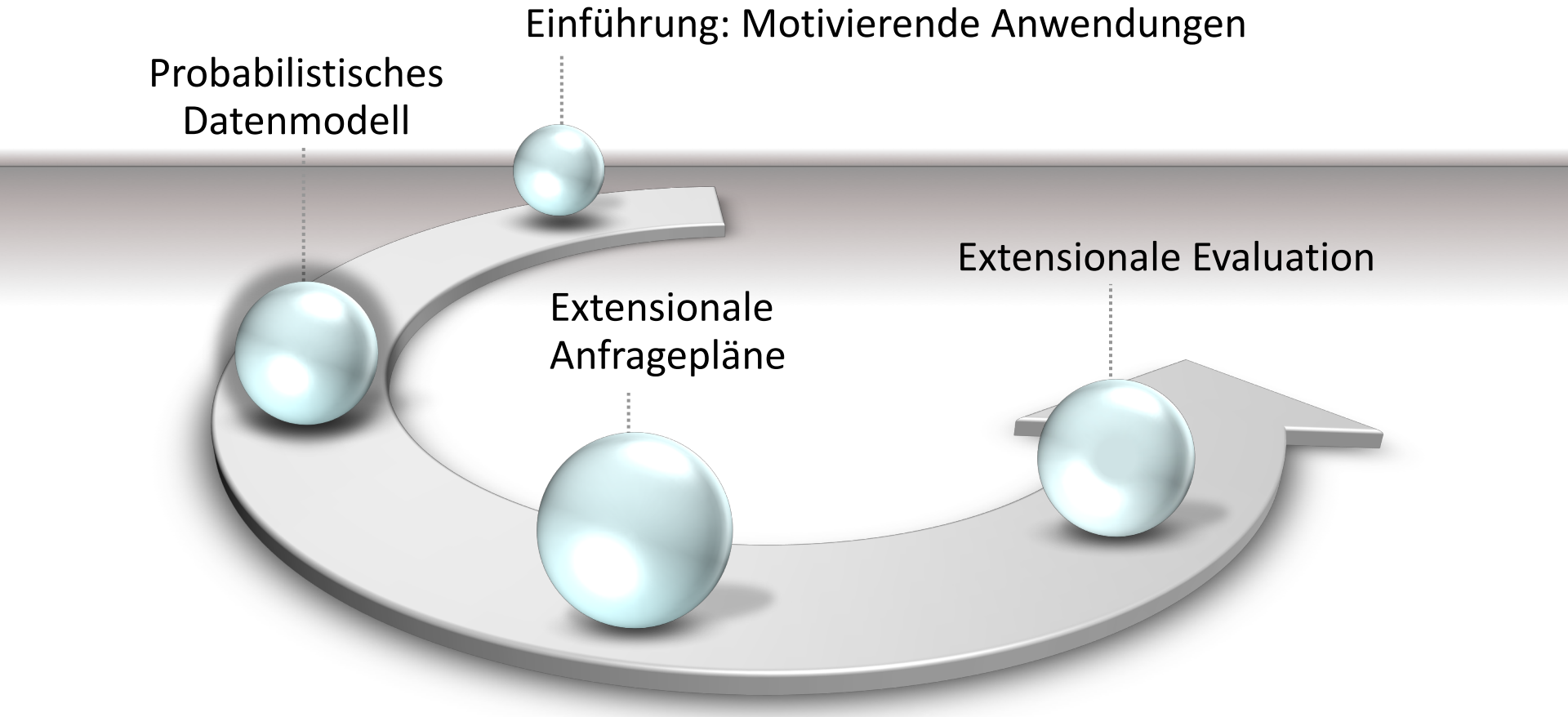
Zusammenfassung der Anwendungen

- Strukturierte, aber unsichere Daten
- Modelliert als probabilistische Daten
- Antworten für SQL queries annotiert mit Wahrscheinlichkeiten

Probabilistische Datenbank:
- Kombination aus Standard-Datenmanagement mit probabilistischer Inferenz

Non-Standard-Datenbanken

Probabilistische Datenbanken



Non-Standard-Datenbanken

Probabilistische Datenbanken

Prof. Dr. Ralf Möller

Universität zu Lübeck

Institut für Informationssysteme

Wiederholung: Relationales Datenmodell

Daten:
gespeichert in Relationen (= Tabellen)

Owner

Name	Object
Joe	Book302
Joe	Laptop77
Jim	Laptop77
Fred	GgleGlass

Location

Object	Time	Loc
Laptop77	5:07	Hall
Laptop77	9:05	Office
Book302	8:18	Office

Wiederholung: Relationales Datenmodell

Daten:
gespeichert in Relationen (= Tabellen)

Owner

Name	Object
Joe	Book302
Joe	Laptop77
Jim	Laptop77
Fred	GgleGlass

Location

Object	Time	Loc
Laptop77	5:07	Hall
Laptop77	9:05	Office
Book302	8:18	Office

Anfragen: SQL,

Find all owners of objects in the Office

```
-- SQL: z.B. Postgres

SELECT DISTINCT Owner.name
FROM Owner, Location
WHERE Owner.object = Location.object
    and  Location.loc = 'Office'
```

Wiederholung: Relationales Datenmodell

Daten:
gespeichert in Relationen (= Tabellen)

Owner

Name	Object
Joe	Book302
Joe	Laptop77
Jim	Laptop77
Fred	GgleGlass

Location

Object	Time	Loc
Laptop77	5:07	Hall
Laptop77	9:05	Office
Book302	8:18	Office

Anfragen: SQL,

Vereinigung konjunktiver Anfragen
Unions of Conjunctive Queries (UCQs)

Find all owners of objects in the Office

```
-- SQL: z.B. Postgres

SELECT DISTINCT Owner.name
FROM Owner, Location
WHERE Owner.object = Location.object
    and  Location.loc = 'Office'
```

Q(z) = Owner(z,x), Location(x,t,y),y='Office'

NB x,t sind existenzquantifiziert:

Q(z) = ∃x ∃t (Owner(z,x), Location(x,t,'Office'))

Wiederholung: Relationales Datenmodell

Daten:
gespeichert in Relationen (= Tabellen)

Owner

Name	Object
Joe	Book302
Joe	Laptop77
Jim	Laptop77
Fred	GgleGlass

Location

Object	Time	Loc
Laptop77	5:07	Hall
Laptop77	9:05	Office
Book302	8:18	Office

Anfragen: SQL,

Vereinigung konjunktiver Anfragen
Unions of Conjunctive Queries (UCQs)

Find all owners of objects in the Office

```
-- SQL: z.B. Postgres

SELECT DISTINCT Owner.name
FROM Owner, Location
WHERE Owner.object = Location.object
    and  Location.loc = 'Office'
```

$Q(z) = Owner(z,x), Location(x,t,y), y='Office'$

NB x,t sind existenzquantifiziert:

$Q(z) = \exists x\, \exists t\, (Owner(z,x), Location(x,t,'Office'))$

Antwort: Q=

Name
Joe
Jim

Wiederholung: Relationales Datenmodell

Daten:
gespeichert in Relationen (= Tabellen)

Owner

Name	Object
Joe	Book302
Joe	Laptop77
Jim	Laptop77
Fred	GgleGlass

Location

Object	Time	Loc
Laptop77	5:07	Hall
Laptop77	9:05	Office
Book302	8:18	Office

Anfragen: SQL,

Vereinigung konjunktiver Anfragen
Unions of Conjunctive Queries (UCQs)

Find all owners of objects in the Office

```
-- SQL: z.B. Postgres

SELECT DISTINCT Owner.name
FROM Owner, Location
WHERE Owner.object = Location.object
    and  Location.loc = 'Office'
```

$Q(z) = Owner(z,x), Location(x,t,y), y='Office'$

NB x,t sind existenzquantifiziert:

$Q(z) = \exists x\, \exists t\, (Owner(z,x), Location(x,t,'Office'))$

Antwort: Q=

Name
Joe
Jim

Wiederholung: Komplexität der Anfragebeantwortung

Anfrage Q, Datenbank D

- Datenkomplexität:
  fix Q, Komplexität = f(D)

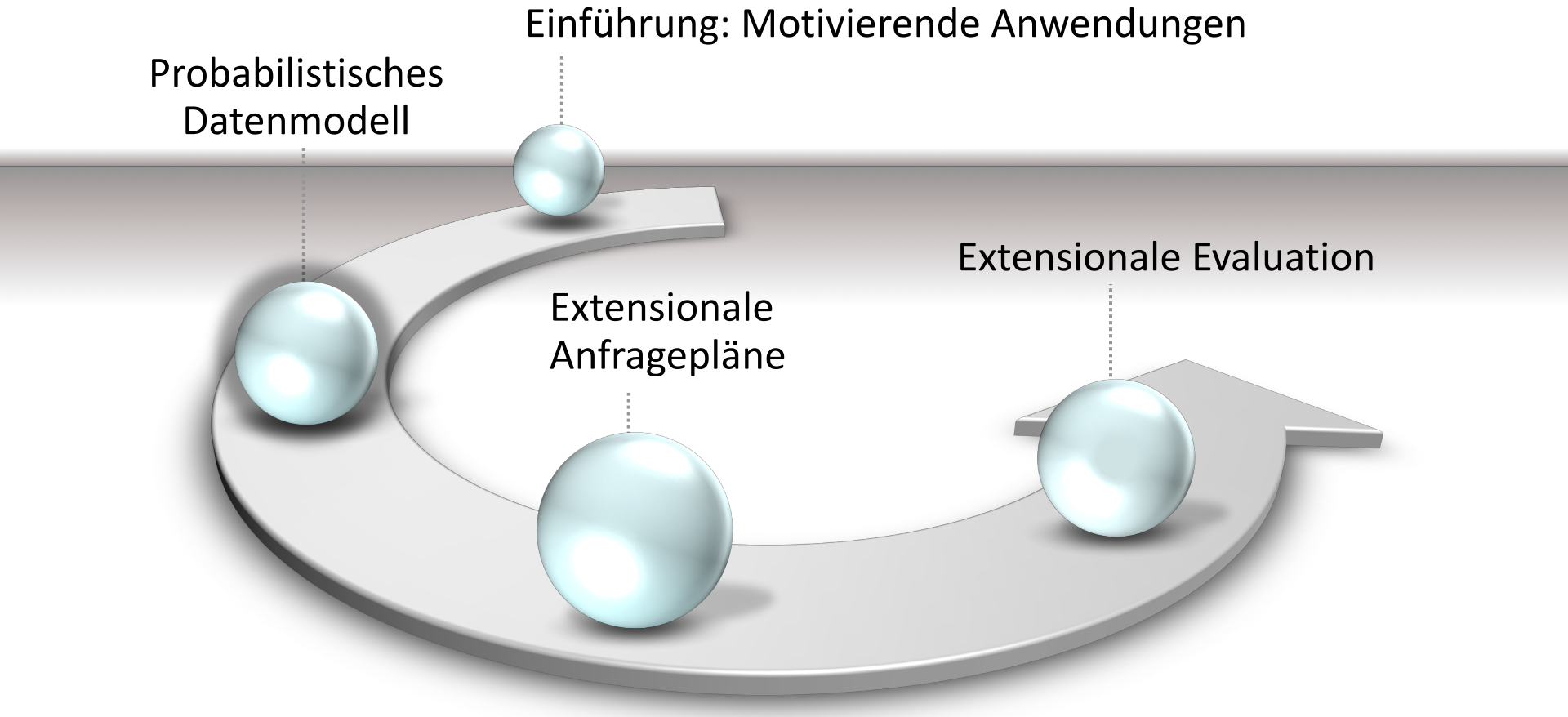
- Anfragekomplexität:
  fix D, Komplexität = f(Q)

- Kombinierte Komplexität: Komplexität = f(D,Q)

Datenkomplexität wird im Bereich der Datenbankforschung betrachtet

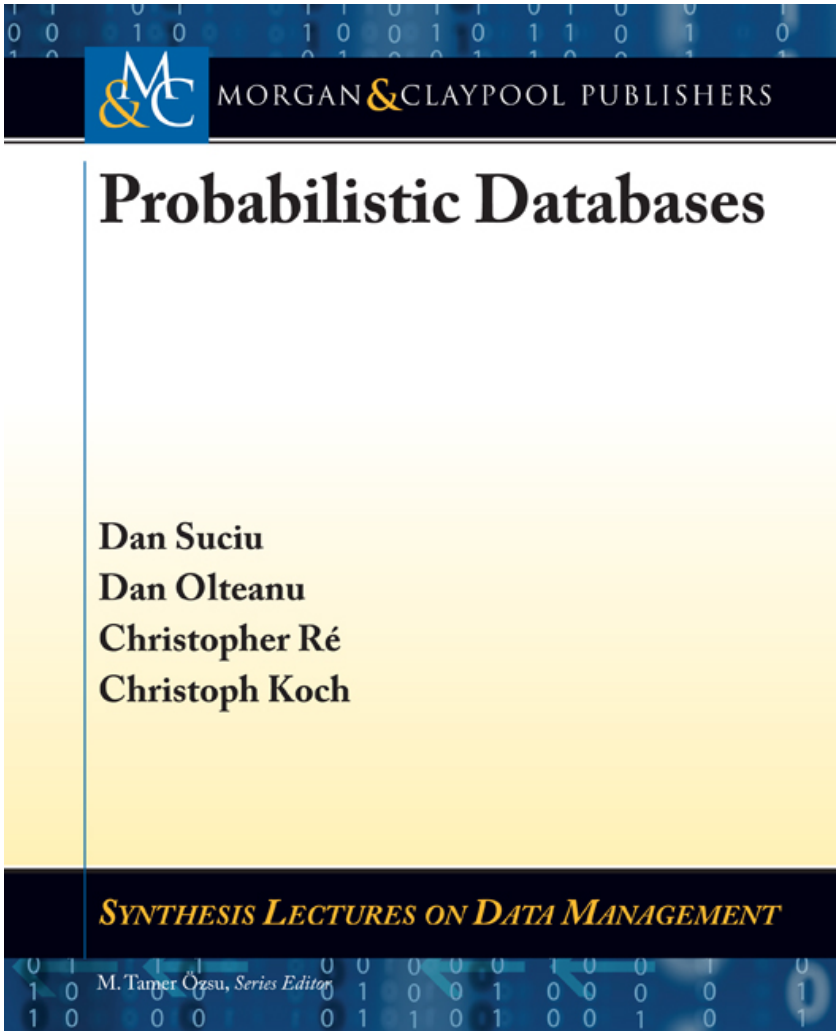
Non-Standard-Datenbanken

Probabilistische Datenbanken



Danksagung

Die Präsentationen sind nach einem Tutorial von Dan Suciu gestaltet und basieren auf dem Lehrbuch Probabilistic Databases



Unvollständige Datenbank

Definition Eine unvollständige Datenbank ist eine endliche Menge von Datenbankinstanzen
W = $(W_1, W_2, \ldots, W_n)$

Jedes W_i heißt mögliche Welt

Unvollständige Datenbank

Definition Eine unvollständige Datenbank ist eine endliche Menge von Datenbankinstanzen
W = $(W_1, W_2, \ldots, W_n)$

Jedes W_i heißt mögliche Welt

W_1	W_2	W_3	W_4
Owner Location			

Owner

Name	Object
Joe	Book302
Joe	Laptop77
Jim	Laptop77
Fred	GgleGlass

Location

Object	Time	Loc
Laptop77	5:07	Hall
Laptop77	9:05	Office
Book302	8:18	Office

Unvollständige Datenbank

Definition Eine unvollständige Datenbank ist eine endliche Menge von Datenbankinstanzen
W = $(W_1, W_2, \ldots, W_n)$

Jedes W_i heißt mögliche Welt

W_1

Owner

Name	Object
Joe	Book302
Joe	Laptop77
Jim	Laptop77
Fred	GgleGlass

Location

Object	Time	Loc
Laptop77	5:07	Hall
Laptop77	9:05	Office
Book302	8:18	Office

W_2

Owner

Name	Object
Joe	Book302
Jim	Laptop77
Fred	GgleGlass

Location

Object	Time	Loc
Book302	8:18	Office

W_3

W_4

Unvollständige Datenbank

Definition Eine unvollständige Datenbank ist eine endliche Menge von Datenbankinstanzen
W = $(W_1, W_2, \ldots, W_n)$

Jedes W_i heißt mögliche Welt

W_1

Owner

Name	Object
Joe	Book302
Joe	Laptop77
Jim	Laptop77
Fred	GgleGlass

Location

Object	Time	Loc
Laptop77	5:07	Hall
Laptop77	9:05	Office
Book302	8:18	Office

W_2

Owner

Name	Object
Joe	Book302
Jim	Laptop77
Fred	GgleGlass

Location

Object	Time	Loc
Book302	8:18	Office

W_3

Owner

Name	Object
Jim	Laptop77

Location

Object	Time	Loc
Laptop77	5:07	Hall
Laptop77	9:05	Office

W_4

Owner

Name	Object
Joe	Book302
Jim	Laptop77
Fred	GgleGlass

Location

Object	Time	Loc
Laptop77	5:07	Hall
Laptop77	9:05	Office
Book302	8:18	Office

Unvollständige Datenbank: Anfragesemantik

Definition Gegeben eine Anfrage Q, eine unvollständige DB **W**:

- Eine Antwort t ist sicher (certain), falls $\forall W_i$, $t \in Q(W_i)$
- Eine Antwort t ist möglich (possible) falls $\exists W_i$, $t \in Q(W_i)$

Unvollständige Datenbank: Anfragesemantik

Definition Gegeben eine Anfrage Q, eine unvollständige DB **W**:

- Eine Antwort t ist sicher (certain), falls $\forall W_i$, $t \in Q(W_i)$
- Eine Antwort t ist möglich (possible) falls $\exists W_i$, $t \in Q(W_i)$

Q(z) = Owner(z,x),
Location(x,t,'Office')

W_1

Owner

Name	Object
Joe	Book302
Joe	Laptop77
Jim	Laptop77
Fred	GgleGlass

Location

Object	Time	Loc
Laptop77	5:07	Hall
Laptop77	9:05	Office
Book302	8:18	Office

W_2

Owner

Name	Object
Joe	Book302
Jim	Laptop77
Fred	GgleGlass

Location

Object	Time	Loc
Book302	8:18	Office

W_3

Owner

Name	Object
Joe	Laptop77

Location

Object	Time	Loc
Laptop77	5:07	Hall
Laptop77	9:05	Office

W_4

Owner

Name	Object
Joe	Book302
Jim	Laptop77
Fred	GgleGlass

Location

Object	Time	Loc
Laptop77	5:07	Hall
Laptop77	9:05	Office
Book302	8:18	Office

Unvollständige Datenbank: Anfragesemantik

Definition Gegeben eine Anfrage Q, eine unvollständige DB **W**:

- Eine Antwort t ist sicher (certain), falls $\forall W_i$, $t \in Q(W_i)$
- Eine Antwort t ist möglich (possible) falls $\exists W_i$, $t \in Q(W_i)$

Q(z) = Owner(z,x),
Location(x,t,'Office')

W_1

Owner

Name	Object
Joe	Book302
Joe	Laptop77
Jim	Laptop77
Fred	GgleGlass

Location

Object	Time	Loc
Laptop77	5:07	Hall
Laptop77	9:05	Office
Book302	8:18	Office

Q=

Joe
Jim

W_2

Owner

Name	Object
Joe	Book302
Jim	Laptop77
Fred	GgleGlass

Location

Object	Time	Loc
Book302	8:18	Office

Q=

Joe

W_3

Owner

Name	Object
Joe	Laptop77

Location

Object	Time	Loc
Laptop77	5:07	Hall
Laptop77	9:05	Office

Q=

Joe

W_4

Owner

Name	Object
Joe	Book302
Jim	Laptop77
Fred	GgleGlass

Location

Object	Time	Loc
Laptop77	5:07	Hall
Laptop77	9:05	Office
Book302	8:18	Office

Q=

Joe
Jim

Unvollständige Datenbank: Anfragesemantik

Definition Gegeben eine Anfrage Q, und eine unvollständige Datenbasis **W**:

- Eine Antwort t heißt sicher (certain), falls $\forall W_i$: $t \in Q(W_i)$
- Eine Antwort t heißt possible, falls $\exists W_i$: $t \in Q(W_i)$

Q(z) = Owner(z,x),
Location(x,t,'Office')

Sichere Antwort für Q: Joe
Mögliche Antwort f. Q: Joe, Jim

W_1

Owner

Name	Object
Joe	Book302
Joe	Laptop77
Jim	Laptop77
Fred	GgleGlass

Location

Object	Time	Loc
Laptop77	5:07	Hall
Laptop77	9:05	Office
Book302	8:18	Office

Q=

Joe
Jim

W_2

Owner

Name	Object
Joe	Book302
Jim	Laptop77
Fred	GgleGlass

Location

Object	Time	Loc
Book302	8:18	Office

Q=

Joe

W_3

Owner

Name	Object
Joe	Laptop77

Location

Object	Time	Loc
Laptop77	5:07	Hall
Laptop77	9:05	Office

Q=

Joe

W_4

Owner

Name	Object
Joe	Book302
Jim	Laptop77
Fred	GgleGlass

Location

Object	Time	Loc
Laptop77	5:07	Hall
Laptop77	9:05	Office
Book302	8:18	Office

Q=

Joe
Jim

Probabilistische Datenbank

Definition Eine probabilistische DB ist ein Tupel (**W**, P), wobei **W** eine unvollständige DB und P: **W** → [0,1] eine Wahrscheinlichkeitsverteilung ist: $\Sigma_{i=1,n} P(W_i) = 1$

Probabilistische Datenbank: Distribution Semantics

Definition Eine probabilistische DB ist ein Tupel (**W**, P), wobei **W** eine unvollständige DB und P: **W** → [0,1] eine Wahrscheinlichkeitsverteilung ist: $\Sigma_{i=1,n} P(W_i) = 1$

W_1 — 0.3

Owner

Name	Object
Joe	Book302
Joe	Laptop77
Jim	Laptop77
Fred	GgleGlass

Location

Object	Time	Loc
Laptop77	5:07	Hall
Laptop77	9:05	Office
Book302	8:18	Office

W_2 — 0.4

Owner

Name	Object
Joe	Book302
Jim	Laptop77
Fred	GgleGlass

Location

Object	Time	Loc
Book302	8:18	Office

W_3 — 0.2

Owner

Name	Object
Jim	Laptop77

Location

Object	Time	Loc
Laptop77	5:07	Hall
Laptop77	9:05	Office

W_4 — 0.1

Owner

Name	Object
Joe	Book302
Jim	Laptop77
Fred	GgleGlass

Location

Object	Time	Loc
Laptop77	5:07	Hall
Laptop77	9:05	Office
Book302	8:18	Office

Sato, T.: A statistical learning method for logic programs with distribution semantics. Proc. of the 12th Int. Conf. on Logic Programming (ICLP95), Tokyo, pp.715–729, 1995.

Probabilistische Datenbank: Anfragesemantik

Definition Gegeben eine Anfrage Q, eine probabilistische DB (**W**,P):

- Die Randwahrscheinlichkeit einer Antwort t ist:
  $P(t) = \Sigma \ \{ P(W_i) \mid W_i \in \mathbf{W}, \ t \in Q(W_i) \}$

Probabilistische Datenbank: Anfragesemantik

Definition Gegeben eine Anfrage Q, eine probabilistische DB (**W**,P):

- Die Randwahrscheinlichkeit einer Antwort t ist:
  $P(t) = \Sigma \ \{ P(W_i) \mid W_i \in \mathbf{W}, \ t \in Q(W_i) \}$

Q(z) = Owner(z,x),
Location(x,t,'Office')

W_1 — 0.3

Owner

Name	Object
Joe	Book302
Joe	Laptop77
Jim	Laptop77
Fred	GgleGlass

Location

Object	Time	Loc
Laptop77	5:07	Hall
Laptop77	9:05	Office
Book302	8:18	Office

W_2 — 0.4

Owner

Name	Object
Joe	Book302
Jim	Laptop77
Fred	GgleGlass

Location

Object	Time	Loc
Book302	8:18	Office

W_3 — 0.2

Owner

Name	Object
Jim	Laptop77

Location

Object	Time	Loc
Laptop77	5:07	Hall
Laptop77	9:05	Office

W_4 — 0.1

Owner

Name	Object
Joe	Book302
Jim	Laptop77
Fred	GgleGlass

Location

Object	Time	Loc
Laptop77	5:07	Hall
Laptop77	9:05	Office
Book302	8:18	Office

Probabilistische Datenbank: Anfragesemantik

Definition Gegeben eine Anfrage Q, eine probabilistische DB (**W**,P):

- Die Randwahrscheinlichkeit einer Antwort t ist:
  $P(t) = \Sigma \ \{ P(W_i) \mid W_i \in \mathbf{W}, \ t \in Q(W_i) \}$

Q(z) = Owner(z,x),
Location(x,t,'Office')

W_1 — 0.3

Owner

Name	Object
Joe	Book302
Joe	Laptop77
Jim	Laptop77
Fred	GgleGlass

Location

Object	Time	Loc
Laptop77	5:07	Hall
Laptop77	9:05	Office
Book302	8:18	Office

Q=

Joe
Jim

W_2 — 0.4

Owner

Name	Object
Joe	Book302
Jim	Laptop77
Fred	GgleGlass

Location

Object	Time	Loc
Book302	8:18	Office

Q=

Joe

W_3 — 0.2

Owner

Name	Object
Joe	Laptop77

Location

Object	Time	Loc
Laptop77	5:07	Hall
Laptop77	9:05	Office

Q=

Joe

W_4 — 0.1

Owner

Name	Object
Joe	Book302
Jim	Laptop77
Fred	GgleGlass

Location

Object	Time	Loc
Laptop77	5:07	Hall
Laptop77	9:05	Office
Book302	8:18	Office

Q=

Joe
Jim

Probabilistische Datenbank: Anfragesemantik

Definition Gegeben eine Anfrage Q, eine probabilistische DB (**W**,P):

- Die Randwahrscheinlichkeit einer Antwort t ist:
  $P(t) = \Sigma \ \{ P(W_i) \mid W_i \in \mathbf{W}, \ t \in Q(W_i) \}$

Q(z) = Owner(z,x),
Location(x,t,'Office')

P(Joe) = 1.0
P(Jim) = 0.4

W_1 — 0.3

Owner

Name	Object
Joe	Book302
Joe	Laptop77
Jim	Laptop77
Fred	GgleGlass

Location

Object	Time	Loc
Laptop77	5:07	Hall
Laptop77	9:05	Office
Book302	8:18	Office

Q=

Joe
Jim

W_2 — 0.4

Owner

Name	Object
Joe	Book302
Jim	Laptop77
Fred	GgleGlass

Location

Object	Time	Loc
Book302	8:18	Office

Q=

Joe

W_3 — 0.2

Owner

Name	Object
Joe	Laptop77

Location

Object	Time	Loc
Laptop77	5:07	Hall
Laptop77	9:05	Office

Q=

Joe

W_4 — 0.1

Owner

Name	Object
Joe	Book302
Jim	Laptop77
Fred	GgleGlass

Location

Object	Time	Loc
Laptop77	5:07	Hall
Laptop77	9:05	Office
Book302	8:18	Office

Q=

Joe
Jim

Diskussion

- Intuition: Eine probabilistische Datenbank sagt aus, dass eine Datenbank in einem von verschiedenen möglichen Zuständen ist. Jeder Zustand hat eine Wahrscheintlichkeit

- Mögliche Anfrageantworten: Eine Menge von Antworten, annotiert mit Wahrscheinlichkeiten:

  $(t_1, p_1), (t_2, p_2), (t_3, p_3), \ldots$

  Üblicherweise: $p_1 \geq p_2 \geq p_3 \geq \ldots$

- Problem: Die Anzahl der möglichen Welten in einer probabilistischen Datenbank ist sehr groß.

- Ziel: Anfragebeantwortung ohne explizite Generierung aller möglichen Welten (eventuell Einschränkungen in der Ausdrucksstärke hinnehmen)

Unabhängige und disjunkte Tupel

Definition Gegeben eine probabilistische DB (**W**, P).
Zwei Tupel t_1, t_2 werden heißen:

- unabhängig, falls: $P(t_1\ t_2) = P(t_1)\ P(t_2)$
- disjunkt (or exklusiv), falls: $P(t_1\ t_2) = 0$

Unabhängige und disjunkte Tupel

Definition Gegeben eine probabilistische DB (**W**, P).
Zwei Tupel t_1, t_2 werden heißen:

- unabhängig, falls: $P(t_1\ t_2) = P(t_1)\ P(t_2)$
- disjunkt (or exklusiv), falls: $P(t_1\ t_2) = 0$

Definition Eine probabilistische DB heißt *block-unabhängig-disjunkt* (BUD), falls die Tupel in Blöcke gruppiert werden können, so dass:

- Tupel vom gleichen Block disjunkt sind
- Tupel von verschiedenen Blöcken unabhängig sind.

Beispiel: BUD-Tabelle

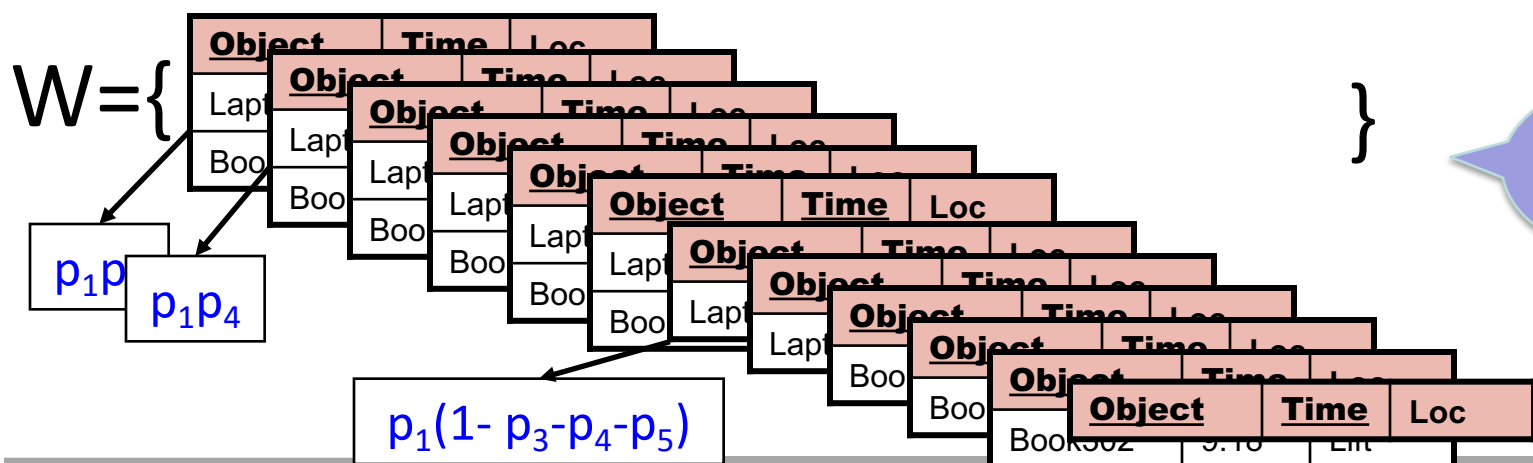
BUD Tabelle

Object	Time	Loc	P
Laptop77	9:07	Rm444	p_1
Laptop77	9:07	Hall	p_2
Book302	9:18	Office	p_3
Book302	9:18	Rm444	p_4
Book302	9:18	Lift	p_5

disjunkt

disjunkt

Unab-
hängig

W={  }

Mögliche
Welten

Das Anfrage-Evaluationsproblem

Gegeben: BUD-Datenbank D, Anfrage Q, Ausgabetupel t

Berechne: P(t)

NB: D habe, sagen wir, 1.000.000 Tupel,
dann ist die Anzahl der möglichen Welten: $2^{1.000.000}$

Herausforderung: Berechne P(t) effizient, in der Größe von D

Datenkomplexität: die Komplexität von P
hängt von D, der Anzahl der Datenelemente, ab.

Ein Beispiel

Boolesche Anfrage:
Join-Tupel vorhanden?

```
SELECT DISTINCT 'true'
FROM R, S
WHERE R.x = S.x
```

Q() = R(x), S(x,y)

$$P(Q) = 1 - \{1 - p1 * [\ 1-(1-q1)*(1-q2)\ ]\} *$$

$$\{1 - p2 * [\ 1-(1-q3)*(1-q4)*(1-q5)]\}$$

Man kann P(Q) in PTIME
bzgl. der Größe der DB D bestimmen

R

x	P
a1	p1
a2	p2
a3	p3

S

x	y	P
a1	b1	q1
a1	b2	q2
a2	b3	q3
a2	b4	q4
a2	b5	q5

Zusammenfassung: Das probabilistische Datenmodell

- Mögliche-Welten-Semantik:
  Mächtig, aber schwierig zu repräsentieren

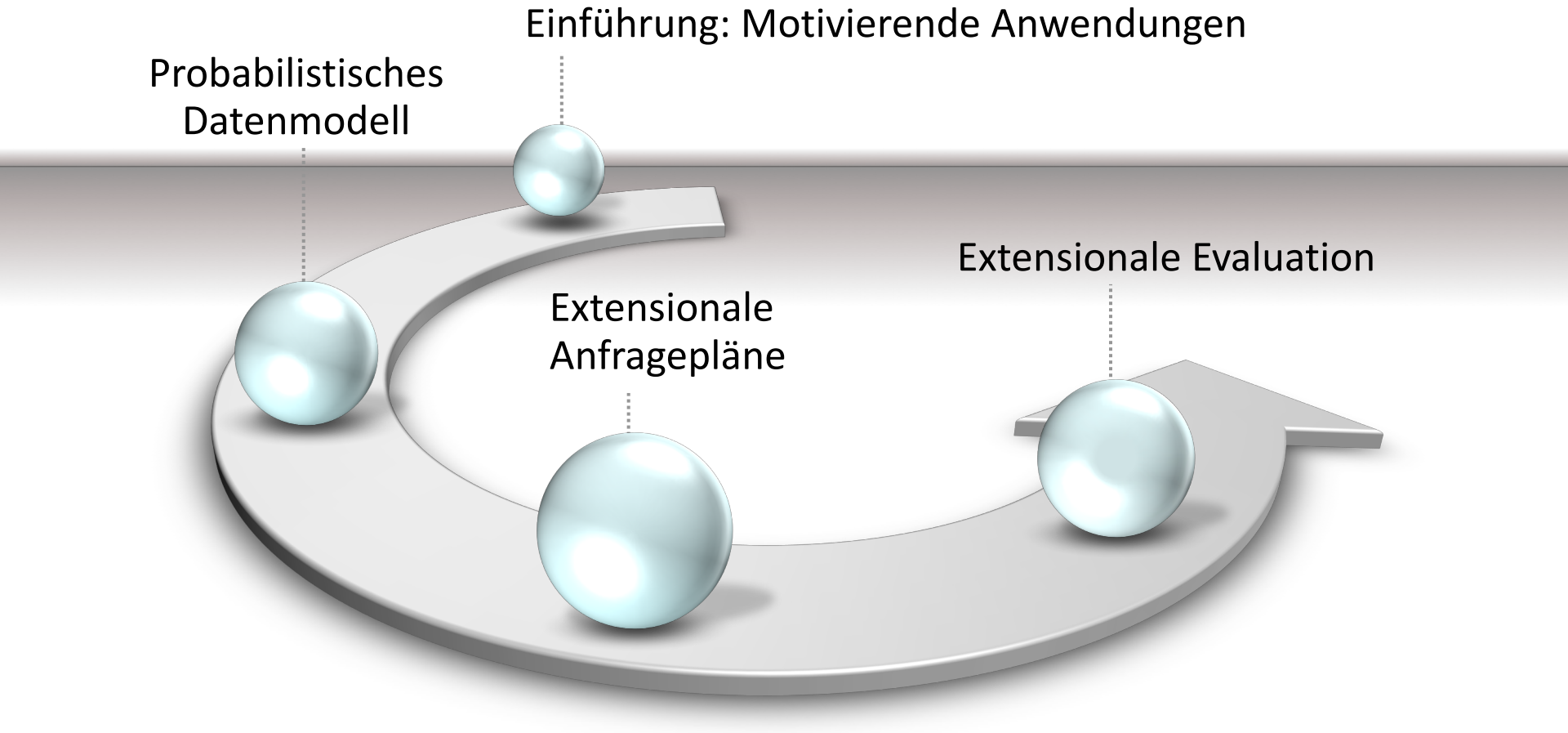
- Block-unabhängig-disjunkte Datenbasen haben effiziente Repräsentationen:
  D wird in traditioneller DB gespeichert

- Unabhängige Datenbasen: noch einfacher

Herausforderung: evaluiere Q effizient
bzgl. der Größe von D

Non-Standard-Datenbanken

Probabilistische Datenbanken



Relationale Algebra

1. Verbund (join) $\bowtie$

2. Projektion (mit Duplikat-Elimination) Π

3. Vereinigung $\cup$

1. Auswahl (selection) σ

2. Differenz: hier nicht verwendet $-$

Wiederholung: Anfragebearbeitungspläne

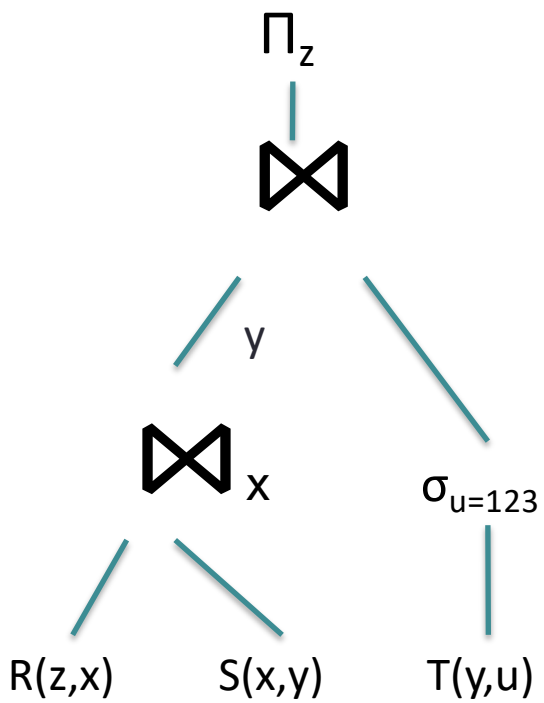
```
SELECT DISTINCT R.z
FROM R, S, T
WHERE R.x = S.x
      and S.y=T.y
      and T.u = 123
```

$Q(z) = R(z,x), S(x,y),T(y,u)$

Wiederholung: Anfragebearbeitungspläne

```
SELECT DISTINCT R.z
FROM R, S, T
WHERE R.x = S.x
      and S.y=T.y
      and T.u = 123
```

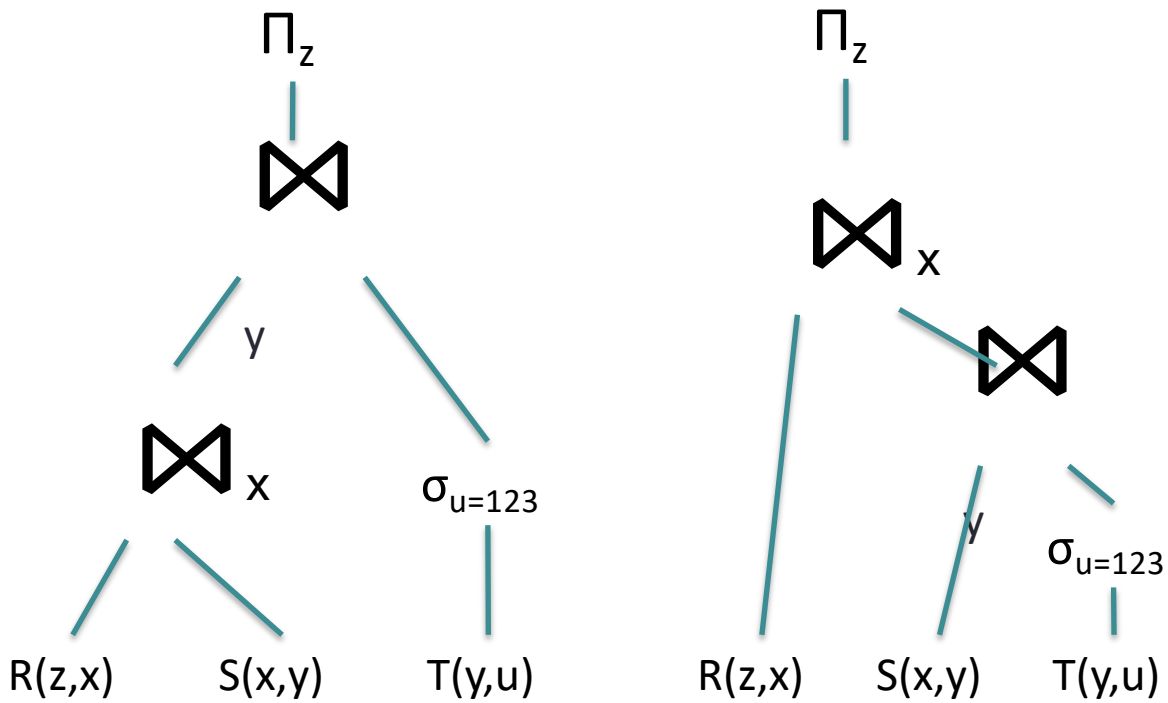
$Q(z) = R(z,x), S(x,y), T(y,u)$



Wiederholung: Anfragebearbeitungspläne

```
SELECT DISTINCT R.z
FROM R, S, T
WHERE R.x = S.x
      and S.y=T.y
      and T.u = 123
```

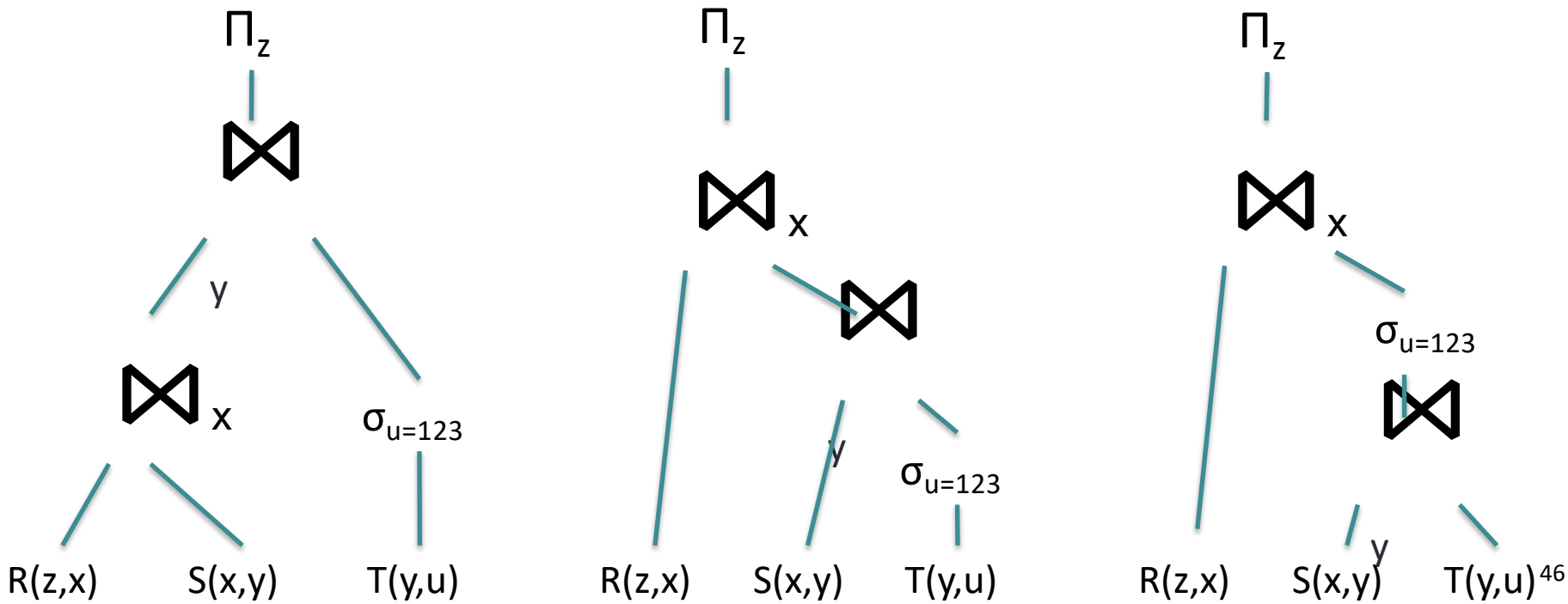
$Q(z) = R(z,x), S(x,y), T(y,u)$



Wiederholung: Anfragebearbeitungspläne

```
SELECT DISTINCT R.z
FROM R, S, T
WHERE R.x = S.x
      and S.y=T.y
      and T.u = 123
```

$Q(z) = R(z,x), S(x,y), T(y,u)$



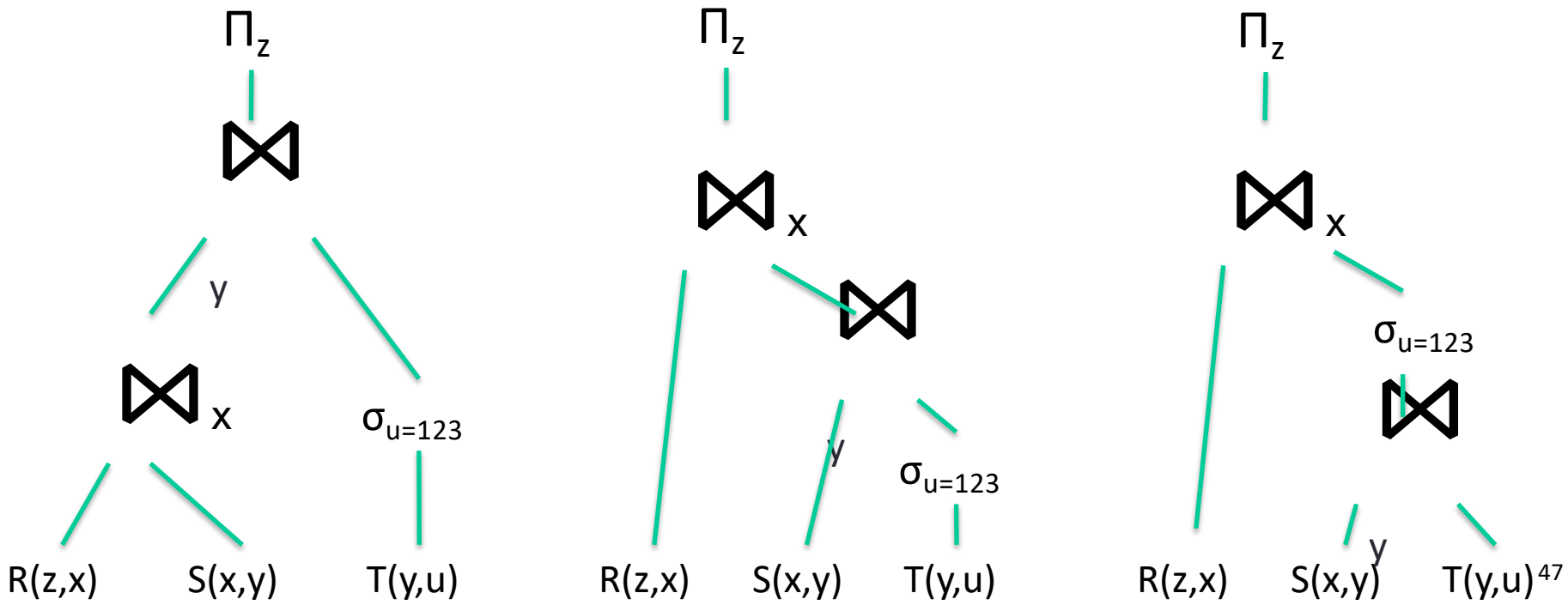
Wiederholung: Anfragebearbeitungspläne

```
SELECT DISTINCT R.z
FROM R, S, T
WHERE R.x = S.x
      and S.y=T.y
      and T.u = 123
```

Q(z) = R(z,x), S(x,y),T(y,u)

Diese Pläne sind äquivalent (liefern gleiche Ergebnisse)

Der Anfrageoptimierer wählt den Plan mit den geschätzt geringsten Kosten



Extensionale Pläne

- Kernidee:
  - Modifiziere jeden Operator, sodass Wahrscheinlichkeiten für die Ausgabe berechnet werden

- Annahmen notwendig:
  - Ereignisse sind
    - unabhängig oder
    - disjunkt (exklusive)

Extensionale Operatoren

Independent join

A	B	P
a1	b1	p1*q1
a1	b2	p1*q2
a2	b3	p2*q3
a2	b4	p2*q4
a2	b5	p2*q5

$\bowtie^{i}$

i für independent

R(A)

A	P
a1	p1
a2	p2
a3	p3

S(A,B)

A	B	P
a1	b1	q1
a1	b2	q2
a2	b3	q3
a2	b4	q4
a2	b5	q5

Extensionale Operatoren

Independent join

A	B	P
a1	b1	p1*q1
a1	b2	p1*q2
a2	b3	p2*q3
a2	b4	p2*q4
a2	b5	p2*q5

$\bowtie^{i}$

i für independent

R(A)

A	P
a1	p1
a2	p2
a3	p3

S(A,B)

A	B	P
a1	b1	q1
a1	b2	q2
a2	b3	q3
a2	b4	q4
a2	b5	q5

Independent project

A	P
a1	1 - (1-q1)*(1-q2)
a2	1 - (1-q3)*(1-q4)*(1-q5)

Π^{i}_{A}

i für independent

S(A,B)

A	B	P
a1	b1	q1
a1	b2	q2
a2	b3	q3
a2	b4	q4
a2	b5	q5

Extensionale Operatoren

Independent join

A	B	P
a1	b1	p1*q1
a1	b2	p1*q2
a2	b3	p2*q3
a2	b4	p2*q4
a2	b5	p2*q5

$\bowtie^{i}$

i für independent

R(A)

A	P
a1	p1
a2	p2
a3	p3

S(A,B)

A	B	P
a1	b1	q1
a1	b2	q2
a2	b3	q3
a2	b4	q4
a2	b5	q5

Independent project

A	P
a1	1 - (1-q1)*(1-q2)
a2	1 - (1-q3)*(1-q4)*(1-q5)

Π^{i}_{A}

i für independent

S(A,B)

A	B	P
a1	b1	q1
a1	b2	q2
a2	b3	q3
a2	b4	q4
a2	b5	q5

Selection

A	B	P
a2	b2	q3
a2	b3	q4
a2	b2	q5

$\sigma_{A=a2}$

S(A,B)

A	B	P
a1	b1	q1
a1	b1	q2
a2	b2	q3
a2	b3	q4
a2	b2	q5

```
SELECT DISTINCT 'true'
FROM R, S
WHERE R.x = S.x
```

Q() = R(x), S(x,y)

$$P(Q) = 1 - [1-p_1*(1-(1-q_1)*(1-q_2))] *[1- p_2*(1-(1-q_3)*(1-q_4)*(1-q_5))]$$

Beispiel

R

x	P
a_1	p_1
a_2	p_2
a_3	p_3

S

x	y	P
a_1	b_1	q_1
a_1	b_2	q_2
a_2	b_3	q_3
a_2	b_4	q_4
a_2	b_5	q_5

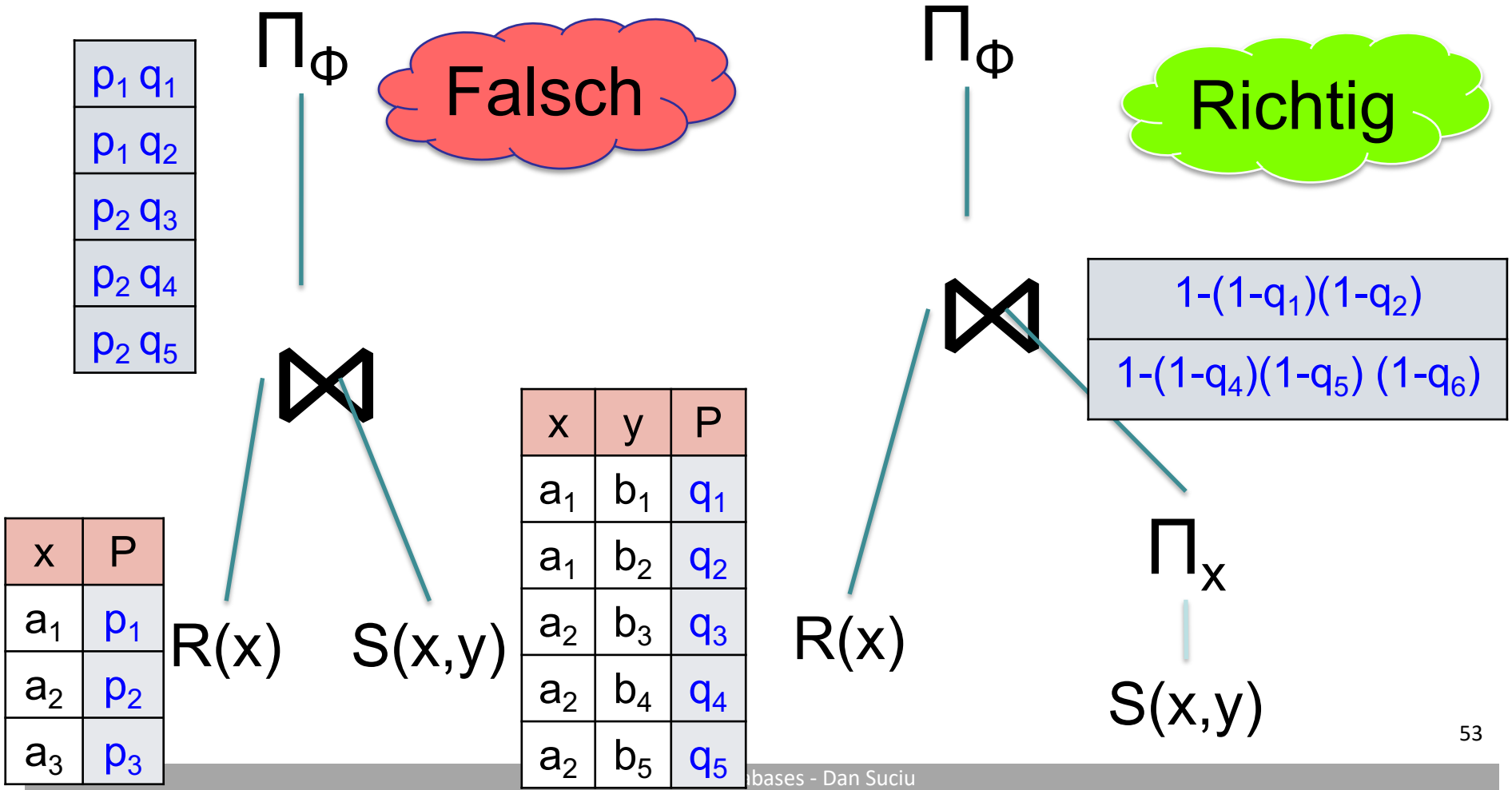
```
SELECT DISTINCT 'true'
FROM R, S
WHERE R.x = S.x
```

$Q() = R(x), S(x,y)$

$P(Q) = 1 - [1-p_1*(1-(1-q_1)*(1-q_2))]$
$*[1- p_2*(1-(1-q_3)*(1-q_4)*(1-q_5))]$

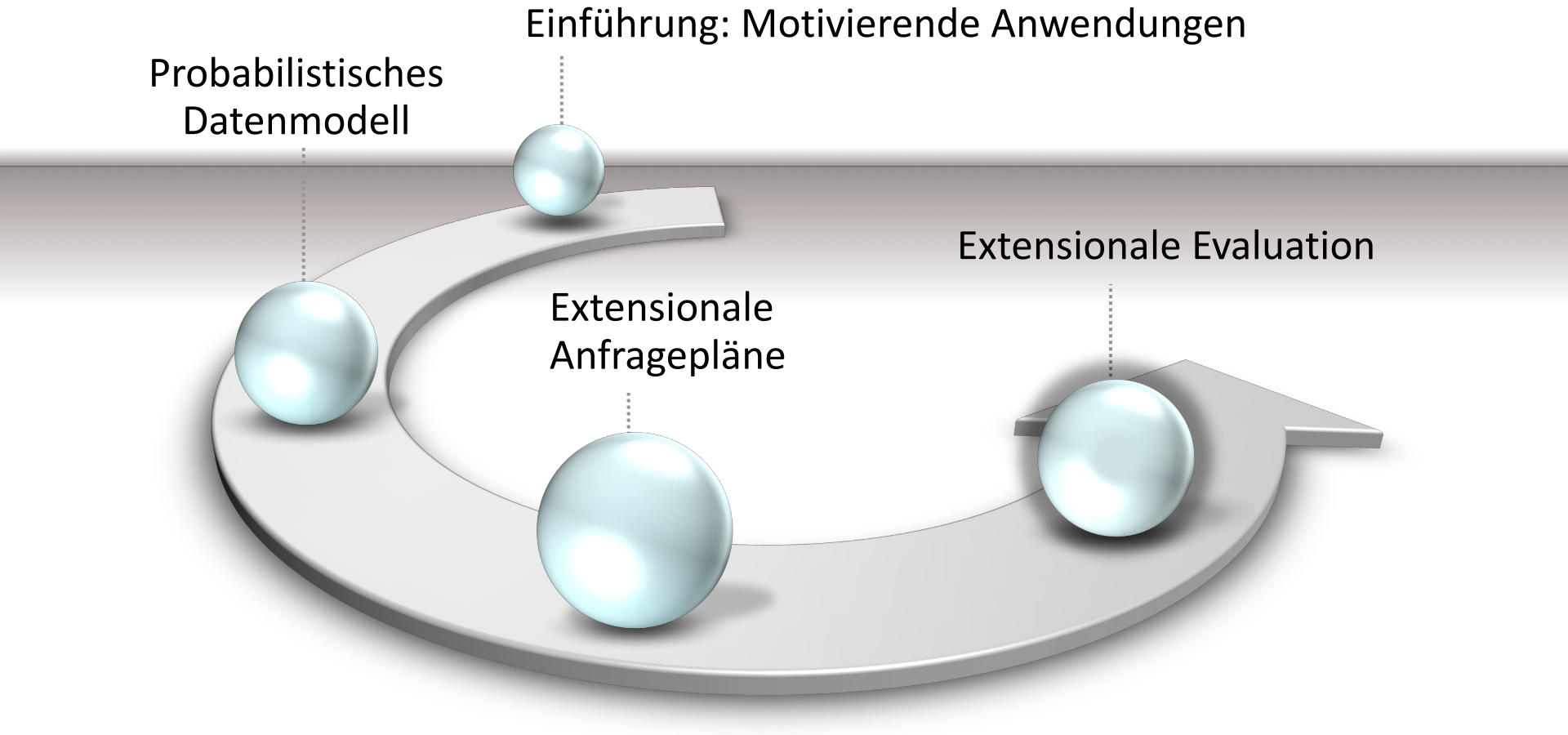
$1-(1-p_1q_1)(1-p_1q_2)(1-p_2q_3)(1-p_2q_4)(1-p_2q_5)$

$1-\{1-p_1[1-(1-q_1)(1-q_2)]\}*$
$\{1-p_2[1-(1-q_4)(1-q_5)\ (1-q_6)]\}$



Non-Standard-Datenbanken

Probabilistische Datenbanken



Extensionale Operatoren

Independent join

A	B	P
a1	b1	p1*q1
a1	b2	p1*q2
a2	b3	p2*q3
a2	b4	p2*q4
a2	b5	p2*q5

$\bowtie^{i}$

i für independent

R(A)

A	P
a1	p1
a2	p2
a3	p3

S(A,B)

A	B	P
a1	b1	q1
a1	b2	q2
a2	b3	q3
a2	b4	q4
a2	b5	q5

Independent project

A	P
a1	1 - (1-q1)*(1-q2)
a2	1 - (1-q3)*(1-q4)*(1-q5)

Π^{i}_{A}

i für independent

S(A,B)

A	B	P
a1	b1	q1
a1	b2	q2
a2	b3	q3
a2	b4	q4
a2	b5	q5

Selection

A	B	P
a2	b2	q3
a2	b3	q4
a2	b2	q5

$\sigma_{A=a2}$

S(A,B)

A	B	P
a1	b1	q1
a1	b1	q2
a2	b2	q3
a2	b3	q4
a2	b2	q5

```
SELECT DISTINCT 'true'
FROM R, S
WHERE R.x = S.x
```

$Q() = R(x), S(x,y)$

$P(Q) = 1 - [1-p_1*(1-(1-q_1)*(1-q_2))]$
$*[1- p_2*(1-(1-q_3)*(1-q_4)*(1-q_5))]$

$1-(1-p_1q_1)(1-p_1q_2)(1-p_2q_3)(1-p_2q_4)(1-p_2q_5)$

$1-\{1-p_1[1-(1-q_1)(1-q_2)]\}*$
$\{1-p_2[1-(1-q_4)(1-q_5)\ (1-q_6)]\}$

$p_1\ q_1$
$p_1\ q_2$
$p_2\ q_3$
$p_2\ q_4$
$p_2\ q_5$

Π_Φ

Falsch

$\bowtie$

x	P
a_1	p_1
a_2	p_2
a_3	p_3

R(x)

S(x,y)

x	y	P
a_1	b_1	q_1
a_1	b_2	q_2
a_2	b_3	q_3
a_2	b_4	q_4
a_2	b_5	q_5

Π_Φ

Richtig

$\bowtie$

$1-(1-q_1)(1-q_2)$
$1-(1-q_4)(1-q_5)\ (1-q_6)$

R(x)

Π_x

S(x,y)

Sichere Pläne

- Sei ein Schema für eine probabilist. DB gegeben
  - Relationen tupel-unabhängig oder BUD bei gegebenem Schlüssel

Definition: Ein Plan heißt *sicher*, wenn er die Wahrscheinlichkeiten für die Ausgabe richtig berechnet

- Anfrageoptimierung: Finde kostengünstigen aber sicheren Plan
- Gibt es für jede Anfrage einen sicheren Plan?

Unsichere Anfragen

R

X	P
x1	p1
x2	p2

S

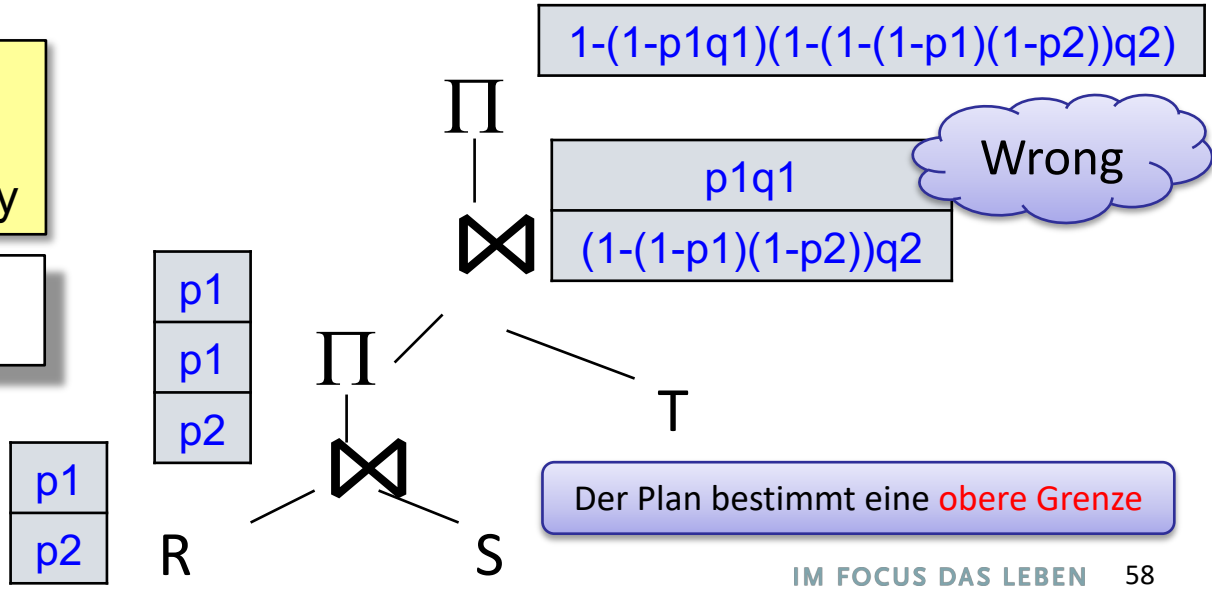
X	Y
x1	y1
x1	y2
x2	y2

T

Y	P
y1	q1
y2	q2

```
SELECT DISTINCT 'yes'
FROM R, S, T
WHERE R.x = S.x and S.y = T.y
```

H_0 :- R(x),S(x,y),T(y)

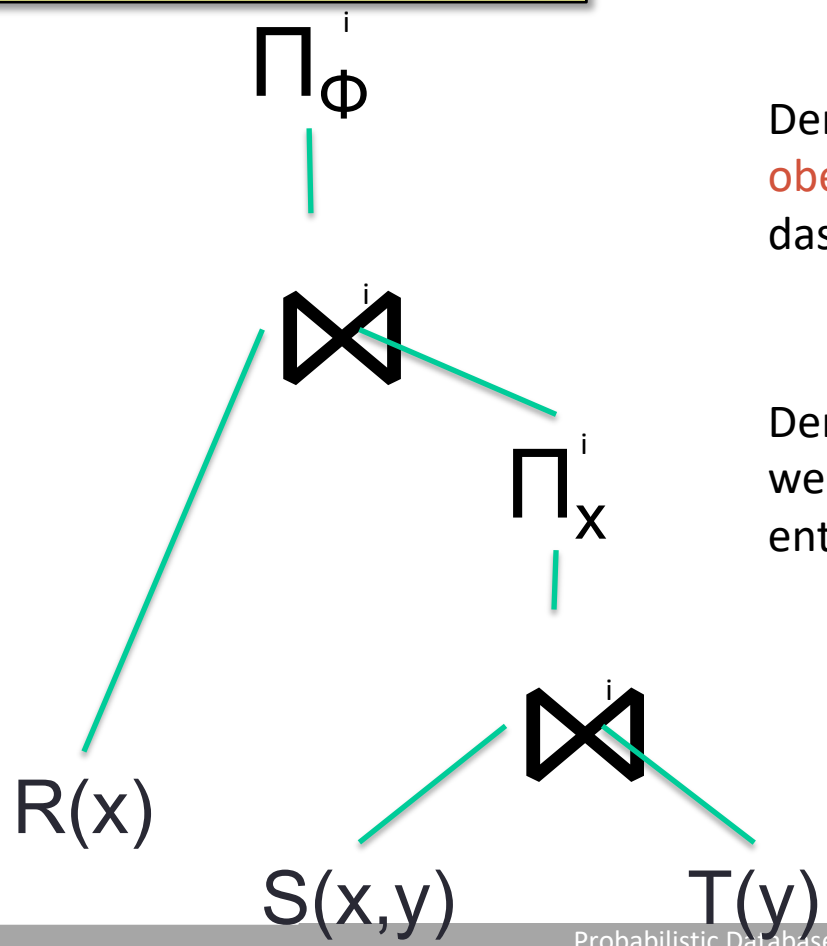


Der Plan bestimmt eine obere Grenze

W Gatterbauer, D Suciu, Oblivious bounds on the probability of Boolean functions, ACM Transactions on Database Systems (TODS) 39 (1), 5, **2013**

Extensionale Pläne in PostgreSQL

```
SELECT DISTINCT 'yes'
FROM R, S, T
WHERE R.x = S.x and S.y = T.y
```



Diese Anfrage ist unsicher.

Der Plan ist unsicher, aber er liefert eine obere Grenze für den Wahrscheinlichkeitswert, dass die Anfrage mit 'yes' beantwortet wird.

Der Plan generiert untere Grenze, wenn die Wahrscheinlichkeiten in T entsprechend angepasst werden

W Gatterbauer, D Suciu, Oblivious bounds on the probability of Boolean functions, ACM Transactions on Database Systems (TODS) 39 (1), 5, **2013**

Untere Grenzen

S

X	Y
x1	y1
x1	y2
x2	y2

$\bowtie_i$

T

Y	P
y1	q1
y2	q2

T'

```
SELECT T.y, 1-exp((ln(1-T.p))/count(*)) AS p
  FROM S,T
  WHERE S.y=T.y
  GROUP by T.y, T.p
```

W Gatterbauer, D Suciu, Oblivious bounds on the probability of Boolean functions, ACM Transactions on Database Systems (TODS) 39 (1), 5, **2013**

Diskussion

- Sichere Anfragen haben einen sicheren Plan und können effektiv berechnet werden

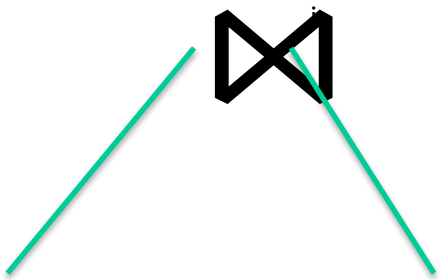
- Für unsichere Anfragen kann kein sicherer Plan bestimmt werden, und es kann gezeigt werden,
  dass sie nicht effizient berechnet werden können

- Jeder extensionale Plan (sicher oder unsicher) kann direkt in SQL ausgedrückt werden – gezeigt am Beispiel von PostgreSQL

Extensionale Pläne in PostgreSQL

A	B	P
a1	b1	p1*q1
a1	b2	p1*q2
a2	b3	p2*q3
a2	b4	p2*q4
a2	b5	p2*q5

```
SELECT R.A, S.B, R.P*S.P
FROM    R, S
WHERE R.A=S.A
```



R(A)

A	P
a1	p1
a2	p2
a3	p3

S(A,B)

A	B	P
a1	b1	q1
a1	b2	q2
a2	b3	q3
a2	b4	q4
a2	b5	q5

Extensionale Pläne in PostgreSQL

A	B	P
a1	b1	p1*q1
a1	b2	p1*q2
a2	b3	p2*q3
a2	b4	p2*q4
a2	b5	p2*q5

```
SELECT R.A, S.B, R.P*S.P
FROM    R, S
WHERE R.A=S.A
```

```
SELECT S.A, 1.0-prod(1.0 - S.p)
FROM    S
GROUP BY S.A
```

A	P
a1	1 - (1-q1)*(1-q2)
a2	1 - (1-q3)*(1-q4)*(1-q5)

$\bowtie^{i}$

Π^{i}_{A}

R(A)

A	P
a1	p1
a2	p2
a3	p3

S(A,B)

A	B	P
a1	b1	q1
a1	b2	q2
a2	b3	q3
a2	b4	q4
a2	b5	q5

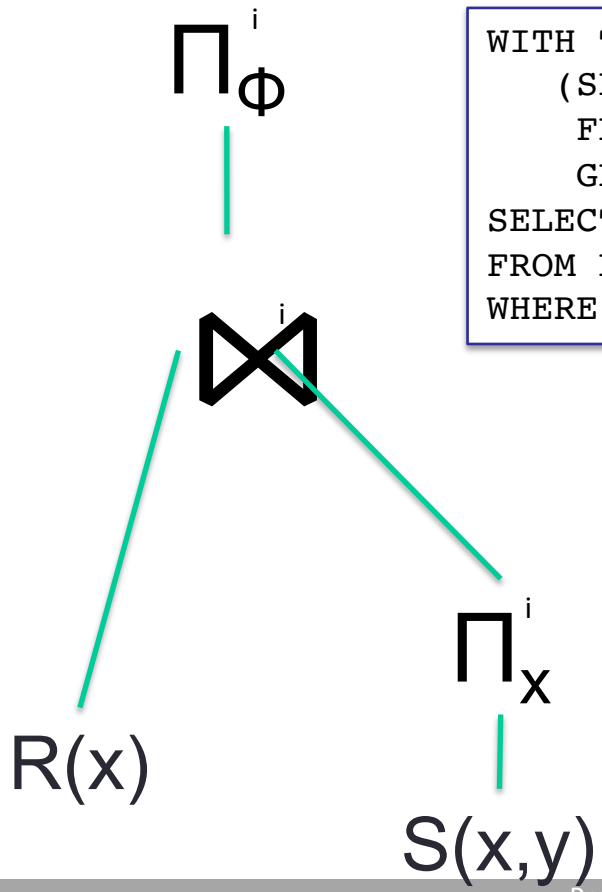
S(A,B)

A	B	P
a1	b1	q1
a1	b2	q2
a2	b3	q3
a2	b4	q4
a2	b5	q5

```
create or replace
  function combine_prod(float, float)
  returns float as
  'select $1 * $2' language SQL;
create or replace
  function final_prod(float)
  returns float as
  'select $1' language SQL;
drop aggregate if exists prod (float);
create aggregate prod(float)
(   sfunc = combine_prod,
    stype = float,
    finalfunc = final_prod,
    initcond = '1.0'
);
```

Extensionale Pläne in PostgreSQL

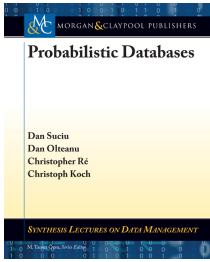
SELECT DISTINCT ‘true’
FROM R, S
WHERE R.x = S.x




```
WITH Temp AS
   (SELECT S.x, 1.0-prod(1.0 - S.p) as p
    FROM S
    GROUP BY S.x)
SELECT ‘true’ as z, 1.0-prod(1.0 – R.P * Temp.P) as p
FROM R, Temp
WHERE R.x = Temp.x
```

Einsichten 2

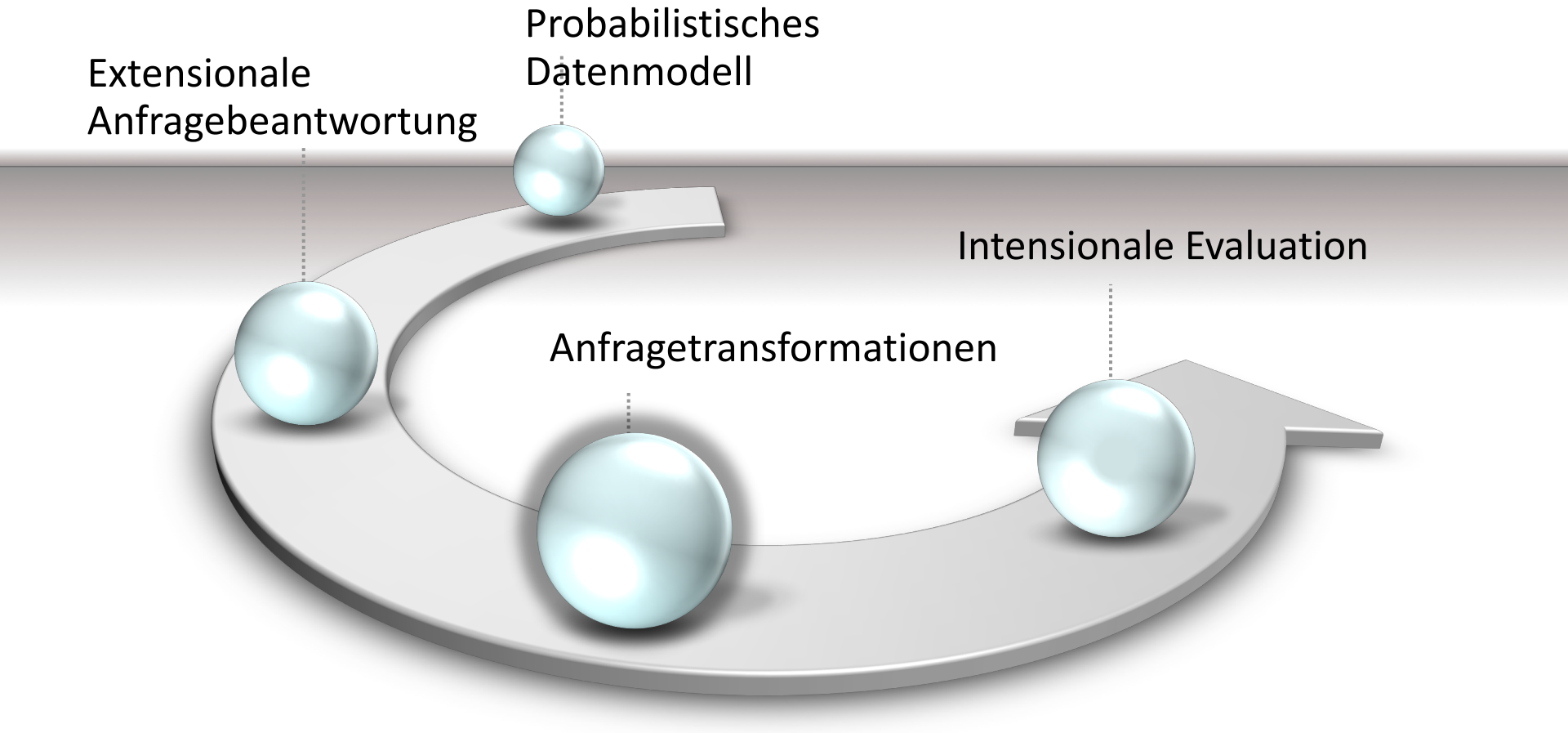
- Man benötigt kein neues probabilistisches DB-System für eine probabilistische Datenbasis, wenn man mit Approximationen leben kann
- Deterministische Approximationen (kein Sampling)



- Wie gut sind die?
- Wir beleuchten später, wie man Top-k-Resultate bestimmen kann (Probability Ranking Principle)

Non-Standard-Datenbanken

Probabilistische Datenbanken



Wiederholung: Unions of Conjunctive Queries

Atom

Owners of items in either “Office444” or “Hall7”:

$$Q(z) = \exists x_1 \exists t_1\ (\text{Owner}(z,x_1) \wedge \text{Location}(x_1,t_1,\text{”Office444”})) \vee$$
$$\exists x_2 \exists t_2\ (\text{Owner}(z,x_2) \wedge \text{Location}(x_2,t_2,\text{”Hall7”}))$$

Ohne Quantoren:

$$Q(z) = \text{Owner}(z,x_1),\text{Location}(x_1,t_1,\text{”Office444”}) \vee \text{Owner}(z,x_2),\text{Location}(x_2,t_2,\text{”Hall7”})$$

Wiederholung: Unions of Conjunctive Queries

Owners of items in either “Office444” or “Hall7”:

$$Q(z) = \exists x_1 \exists t_1\ (\text{Owner}(z,x_1) \wedge \text{Location}(x_1,t_1,\text{”Office444”})) \ \vee$$
$$\exists x_2 \exists t_2\ (\text{Owner}(z,x_2) \wedge \text{Location}(x_2,t_2,\text{”Hall7”}))$$

Ohne Quantoren:

Union of conjunctive queries

$$Q(z) = \text{Owner}(z,x_1),\text{Location}(x_1,t_1,\text{”Office444”}) \ \vee \ \text{Owner}(z,x_2),\text{Location}(x_2,t_2,\text{”Hall7”})$$

Wiederholung: Unions of Conjunctive Queries

Owners of items in either “Office444” or “Hall7”:

$$Q(z) = \exists x_1 \exists t_1\ (\text{Owner}(z,x_1) \wedge \text{Location}(x_1,t_1,\text{”Office444”})) \ \vee$$
$$\exists x_2 \exists t_2\ (\text{Owner}(z,x_2) \wedge \text{Location}(x_2,t_2,\text{”Hall7”}))$$

Ohne Quantoren:

Union of conjunctive queries

$$Q(z) = \text{Owner}(z,x_1),\text{Location}(x_1,t_1,\text{”Office444”}) \ \vee \ \text{Owner}(z,x_2),\text{Location}(x_2,t_2,\text{”Hall7”})$$

Nach Umformung:

$$Q(z) = \text{Owner}(z,x) \wedge \exists t\ [\text{Location}(x,t,\text{”Office444”}) \ \vee \ \text{Location}(x,t,\text{”Hall7”})]$$

Wiederholung: Unions of Conjunctive Queries

Owners of items in either “Office444” or “Hall7”:

$$Q(z) = \exists x_1 \exists t_1\ (\text{Owner}(z,x_1) \wedge \text{Location}(x_1,t_1,\text{”Office444”})) \ \vee$$
$$\exists x_2 \exists t_2\ (\text{Owner}(z,x_2) \wedge \text{Location}(x_2,t_2,\text{”Hall7”}))$$

Ohne Quantoren:

Union of conjunctive queries

$$Q(z) = \text{Owner}(z,x_1),\text{Location}(x_1,t_1,\text{”Office444”}) \ \vee \ \text{Owner}(z,x_2),\text{Location}(x_2,t_2,\text{”Hall7”})$$

Nach Umformung:

$$Q(z) = \text{Owner}(z,x) \wedge \exists t\ [\text{Location}(x,t,\text{”Office444”}) \ \vee \ \text{Location}(x,t,\text{”Hall7”})]$$

Unter Verwendung von:

1. Distributivgesetz für $\vee$, $\wedge$
2. Gesetz für $\exists$, $\vee$: $(\exists x\ P(x)) \vee (\exists y\ T(y)) = \exists z\ (P(z) \vee T(z))$

Vier Regeln, um sichere Anfragen zu erzeugen

- Independent join
- Independent project
- Independent union
- Inclusion/exclusion

"Lifted Inference:" Avoid creating all worlds

Wir beschränken uns auf Boolesche Anfragen.

Regel 1: Independent Join

$$P(Q1 \wedge Q2) = P(Q1)P(Q2)$$

Wenn Q1 und Q2 unabhängig sind
(also keine gemeinsamen Atome haben)

Regel 1: Independent Join

$$P(Q1 \wedge Q2) = P(Q1)P(Q2)$$

Wenn Q1 und Q2 unabhängig sind
(also keine gemeinsamen Atome haben)

Regel 2: Independent Project

$$P(\exists z\ Q) = 1 - \Pi_{a \in Domain}\ (1 - P(Q[a/z])$$

Wenn z eine “Separatorvariable” in Q
ist, also für Konstanten a,b, Q[a/z]
und Q[b/z] unabhängig sind

Regel 1: Independent Join

$$P(Q1 \wedge Q2) = P(Q1)P(Q2)$$

Wenn Q1 und Q2 unabhängig sind
(also keine gemeinsamen Atome haben)

Regel 2: Independent Project

$$P(\exists z\ Q) = 1 - \Pi_{a \in Domain}\ (1 - P(Q[a/z])$$

Wenn z eine "Separatorvariable" in Q
ist, also für Konstanten a,b, Q[a/z]
und Q[b/z] unabhängig sind

Regel 3: Independent Union

$$P(Q1 \vee Q2) = 1 - (1 - P(Q1))(1 - P(Q2))$$

Wenn Q1 und Q2 unabhängig sind
(also keine gemeinsamen Atome haben)

Beispiel

$Q_U = R(x_1),S(x_1,y_1) \vee T(x_2),S(x_2,y_2)$

$= \exists x_1 \exists y_1 R(x_1) \wedge S(x_1,y_1) \vee \exists x_2 \exists y_2 T(x_2) \wedge S(x_2,y_2)$

Beispiel

$Q_U = R(x_1),S(x_1,y_1) \vee T(x_2),S(x_2,y_2)$

$= \exists x_1 \exists y_1 R(x_1) \wedge S(x_1,y_1) \vee \exists x_2 \exists y_2 T(x_2) \wedge S(x_2,y_2)$

$Q_U = \exists z\ [R(z) \wedge S(z,y_1) \vee T(z) \wedge S(z,y_2)]$

Kommutiere $\exists$ mit $\vee$

Beispiel

$Q_U = R(x_1),S(x_1,y_1) \vee T(x_2),S(x_2,y_2)$

$=\exists x_1 \exists y_1 R(x_1) \wedge S(x_1,y_1) \vee \exists x_2 \exists y_2 T(x_2) \wedge S(x_2,y_2)$

$Q_U = \exists z\, [R(z) \wedge S(z,y_1) \vee T(z) \wedge S(z,y_2)]$

Gesetz für $\exists$ mit $\vee$

$P(Q_U) = 1 - \Pi_{a \in Domain}\, (1 - P[R(a) \wedge \underline{S(a,y_1)} \vee T(a) \wedge \underline{S(a,y_2)})]$

Independent project: Für $a \neq b$, sind $Q_U[a/z]$ und $Q_U[b/z]$ unabhängig weil die Atome $R(a),S(a,y_1),T(a),S(a,y_2)$ disjunkt sind von $R(b),S(b,y_1),T(b),S(b,y_2)$

Beispiel

$Q_U = R(x_1),S(x_1,y_1) \vee T(x_2),S(x_2,y_2)$

$=\exists x_1 \exists y_1 R(x_1) \wedge S(x_1,y_1) \vee \exists x_2 \exists y_2 T(x_2) \wedge S(x_2,y_2)$

$Q_U = \exists z\, [R(z) \wedge S(z,y_1) \vee T(z) \wedge S(z,y_2)]$

Gesetz für $\exists$ mit $\vee$

$P(Q_U) = 1 - \Pi_{a \in Domain}\, (1 - P[R(a) \wedge \underline{S(a,y_1)} \vee T(a) \wedge \underline{S(a,y_2)})]$

Independent project: Für $a \neq b$, sind $Q_U[a/z]$ und $Q_U[b/z]$ unabhängig weil die Atome $R(a),S(a,y_1),T(a),S(a,y_2)$ disjunkt sind von $R(b),S(b,y_1),T(b),S(b,y_2)$

$P(Q_U) = 1 - \Pi_{a \in Domain}\, (1 - P[\underline{(R(a) \vee T(a))} \wedge \underline{\exists y.\, S(a,y)}]$

Distribution $\wedge$ über $\vee$

Beispiel

$Q_U = R(x_1),S(x_1,y_1) \vee T(x_2),S(x_2,y_2)$

$= \exists x_1 \exists y_1 R(x_1) \wedge S(x_1,y_1) \vee \exists x_2 \exists y_2 T(x_2) \wedge S(x_2,y_2)$

$Q_U = \exists z\, [R(z) \wedge S(z,y_1) \vee T(z) \wedge S(z,y_2)]$

Kommutiere $\exists$ mit $\vee$

$P(Q_U) = 1 - \Pi_{a \in Domain}\, (1 - P[R(a) \wedge \underline{S(a,y_1)} \vee T(a) \wedge \underline{S(a,y_2)})]$

Independent project: Für $a \neq b$, sind $Q_U[a/z]$ und $Q_U[b/z]$ unabhängig weil die Atome $R(a),S(a,y_1),T(a),S(a,y_2)$ disjunkt sind von $R(b),S(b,y_1),T(b),S(b,y_2)$

$P(Q_U) = 1 - \Pi_{a \in Domain}\, (1 - P[\underline{(R(a) \vee T(a))} \wedge \underline{\exists y.\, S(a,y)}])$

Distribution $\wedge$ über $\vee$

$P(Q_U) = 1 - \Pi_{a \in Domain}\, (1 - P[R(a) \vee T(a)]\; P[\exists y.\, S(a,y)])$

Independent join

Beispiel

$Q_U = R(x_1),S(x_1,y_1) \vee T(x_2),S(x_2,y_2)$

$= \exists x_1 \exists y_1 R(x_1) \wedge S(x_1,y_1) \vee \exists x_2 \exists y_2 T(x_2) \wedge S(x_2,y_2)$

$Q_U = \exists z\, [R(z) \wedge S(z,y_1) \vee T(z) \wedge S(z,y_2)]$

Kommutiere $\exists$ mit $\vee$

$P(Q_U) = 1 - \Pi_{a \in Domain}\, (1 - P[R(a) \wedge \underline{S(a,y_1)} \vee T(a) \wedge \underline{S(a,y_2)})]$

Independent project: Für $a \neq b$, sind $Q_U[a/z]$ und $Q_U[b/z]$ unabhängig weil die Atome $R(a),S(a,y_1),T(a),S(a,y_2)$ disjunkt sind von $R(b),S(b,y_1),T(b),S(b,y_2)$

$P(Q_U) = 1 - \Pi_{a \in Domain}\, (1 - P[\underline{(R(a) \vee T(a))} \wedge \underline{\exists y.\, S(a,y)}]$

Distribution $\wedge$ über $\vee$

$P(Q_U) = 1 - \Pi_{a \in Domain}\, (1 - P[R(a) \vee T(a)]\; P[\exists y.\, S(a,y)]$

Independent join

$P(Q_U) = 1 - \Pi_{a \in Domain}\, (1 - (1-(1-P[R(a)])(1-P[T(a)]))\; (1-\Pi_{b \in Domain}\, (1 - P[S(a,b)])))$

Hierarchische Anfragen

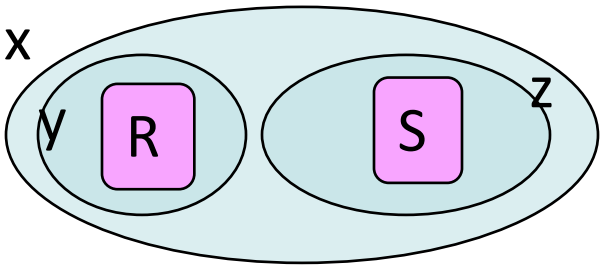
at(x) = Menge von Atomen mit Variable x

Definition Eine Anfrage Q ist hierarchisch falls für alle existentiellen Variablen x, y gilt:

$$at(x) \supseteq at(y) \quad \text{oder} \quad at(y) \supseteq at(x) \quad \text{oder} \quad at(x) \cap at(y) = \emptyset$$

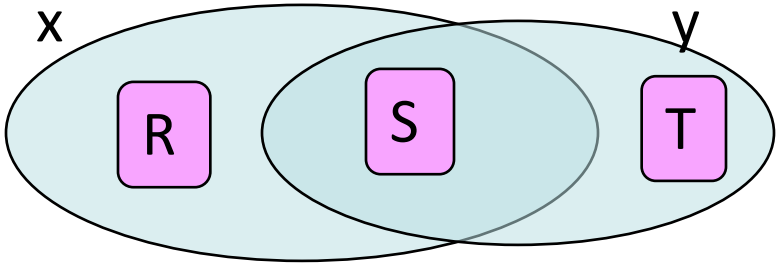
Hierarchisch

Q = R(x,y),S(x,z)



Nicht-hierarchisch

H_0 = R(x), S(x, y), T(y)



Regeln funktionieren für hierarchische Anfragen (Unabhängigkeit)
(Hierarchie ist notwendig, nicht hinreichend für Sicherheit)

N. Dalvi, D. Suciu, The dichotomy of probabilistic inference for unions of conjunctive queries, Journal of the ACM (JACM) 59 (6), 30, **2012**

Warum sind nicht-hierarchische Regeln schwierig?

Lifted Inference Sometimes Fails

$H_0 = \forall x \forall y\, (\text{Smoker}(x) \vee \text{Friend}(x,y) \vee \text{Jogger}(y))$

No rule applies here!

The ∀-rule does not apply, because $H_0[\text{Alice}/x]$ and $H_0[\text{Bob}/x]$ are dependent:

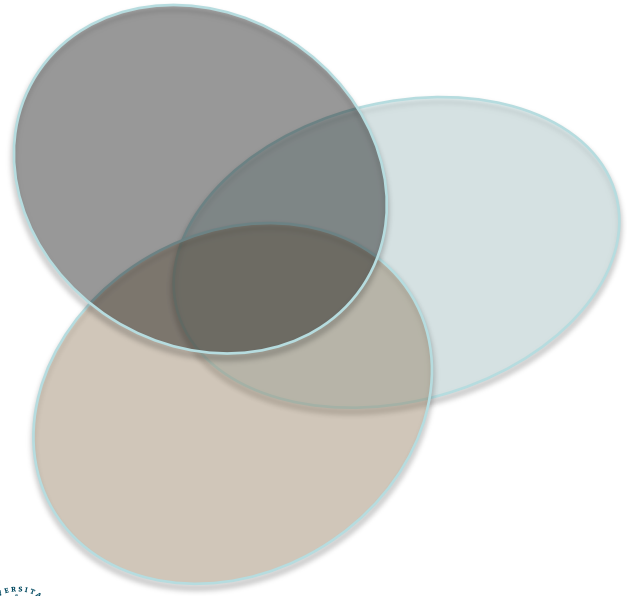
$H_0[\text{Alice}/x] = \forall y\, (\text{Smoker}(\text{Alice}) \vee \text{Friend}(\text{Alice},y) \vee \text{Jogger}(y))$
$H_0[\text{Bob}/x] = \forall y\, (\text{Smoker}(\text{Bob}) \vee \text{Friend}(\text{Bob},y) \vee \text{Jogger}(y))$

Dependent

Theorem. [Dalvi'04] Computing $P(H_0 \mid D)$ is #P-hard in $|D|$

Regel 4: Inclusion-Exclusion

$$P(Q1 \wedge Q2 \wedge Q3) = P(Q1) + P(Q2) + P(Q3) - P(Q1 \vee Q2) - P(Q1 \vee Q3) - P(Q2 \vee Q3) + P(Q1 \vee Q2 \vee Q3)$$



NB: Dieses ist dual zur häufiger verwendeten Formel:

$$P(Q1 \vee Q2 \vee Q3) = P(Q1) + P(Q2) + P(Q3) - P(Q1 \wedge Q2) - P(Q1 \wedge Q3) - P(Q2 \wedge Q3) + P(Q1 \wedge Q2 \wedge Q3)$$

Beispiel

$Q_J = R(x_1), S(x_1,y_1), T(x_2), S(x_2,y_2)$

$= [\exists x_1 \exists y_1 R(x_1) \wedge S(x_1,y_1)] \ \wedge \ [\exists x_2 \exists y_2 T(x_2) \wedge S(x_2,y_2)]$

Beispiel

$Q_J = R(x_1),S(x_1,y_1), T(x_2),S(x_2,y_2)$ $= [\exists x_1 \exists y_1 R(x_1) \wedge S(x_1,y_1)] \wedge [\exists x_2 \exists y_2 T(x_2) \wedge S(x_2,y_2)]$

$Q_J = Q_1 \wedge Q_2$ wobei $Q_1 = R(x_1),S(x_1,y_1)$ $Q_2 = T(x_2),S(x_2,y_2)$

Beispiel

$Q_J = R(x_1),S(x_1,y_1), T(x_2),S(x_2,y_2)$ $= [\exists x_1 \exists y_1 R(x_1) \wedge S(x_1,y_1)] \wedge [\exists x_2 \exists y_2 T(x_2) \wedge S(x_2,y_2)]$

$Q_J = Q_1 \wedge Q_2$ wobei $Q_1 = R(x_1),S(x_1,y_1)$ $Q_2 = T(x_2),S(x_2,y_2)$

$$P(Q_J) = P(Q_1) + P(Q_2) - P(Q_1 \vee Q_2)$$

Q_1 = eine hierarchische CQ ohne Self-Joins

Q_2 = dito

$Q_1 \vee Q_2 = Q_U$, siehe vorige Folien

Einsicht 3

Vereinigung (union) für Self-Joins!

- Conjunctive Queries = Keine “natürliche” Klassen von Anfragen von Probabilistische DBs
- Unions of Conjunctive Queries = die “natürliche” Klasse von Anfragen

Zusammenfassung

- Falls Anfrageformel mit Regeln transformiert werden kann, dann ist das Ergebnis sicher (→ Lifted Inference)
- Regeln sind aber nicht vollständig
  - Suciu et al. stellen noch weitere Regeln vor
    Damit lassen sich alle sicheren Pläne erzeugen
    (inkl. untere Grenze)
- Aber: Nicht alle Anfragen lassen sich mit den Regeln behandeln (→ nicht immer gibt es einen sicheren Plan)
  - Für bestimmte Anfragen müssen potentiell die Welten betrachtet werden (→ Grounded Inference)

L. Antova, T. Jansen, C. Koch, and D. Olteanu. Fast and Simple Relational Processing of Uncertain Data, Proc. 24th International Conference on Data Engineering, ICDE 2008, 983-992, **2008**

Anfragebeantwortung mit Herkunftsformel

- Intensionale Anfrage-beantwortung [FR97]
- Assoziiere mit jedem Tupel ein Ereignis
- Wahrscheinlichkeit, dass Ereignis eintritt / erfüllt ist = Anfragewert

Object	Time	Loc	P	
Laptop77	9:07	Rm444	p_1	e_1
Laptop77	9:07	Hall	p_2	e_2
Book302	9:18	Office	p_3	e_3
Book302	9:18	Rm444	p_4	e_4
Book302	9:18	Lift	p_5	e_5

Anfrageverarbeitung generiert Ereignisausdruck

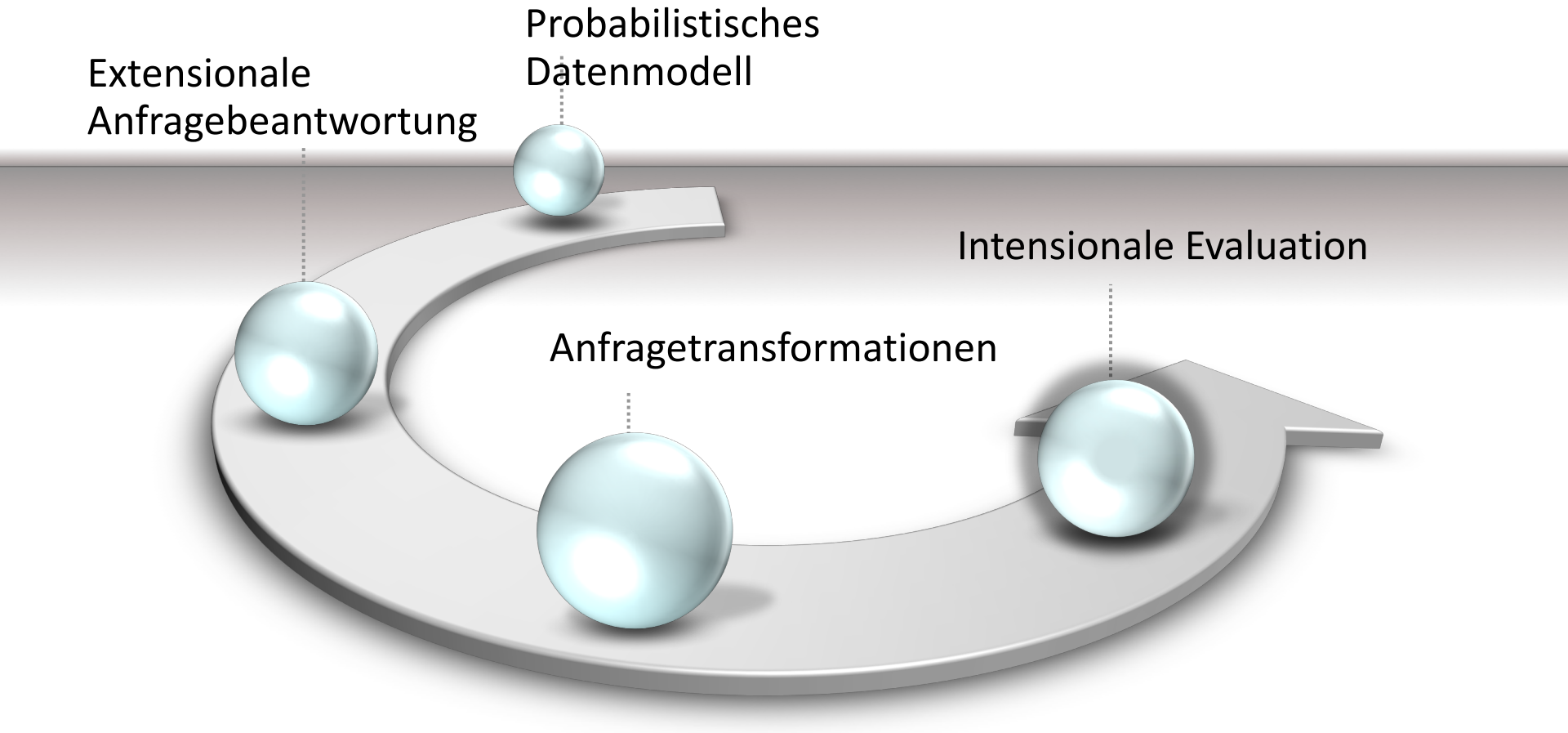
Join	$\times((t_1,[e_1]),(t_2,[e_2]))$	$=((t_1,t_2),[e_1 \wedge e_2])$
Selektion	$\sigma_c((t,[e]))$	$= \begin{cases} (t,[e]) & c(t) = \text{true} \\ \perp & c(t) = \text{false} \end{cases}$
Projektion	$\Pi_A((t_1,[e_1]),\ldots,(t_n,[e_n]))$	$=(t_1[A],[e_1 \vee \cdots \vee e_n])$

Projektion sorgt für DNF

N. Fuhr; T. Rölleke, A Probabilistic Relational Algebra for the Integration of Information Retrieval and Database Systems. ACM Transactions on Information Systems 14(1), **1997**.

Non-Standard-Datenbanken

Probabilistische Datenbanken



Neuere Forschung

- Anwendung von Techniken des
  Weighted Model Counting
  zur Lösung von #SAT-Problemen
- Umwandlung von Wahrscheinlichkeiten in Gewichten

- Folgende Präsentation sind angelehnt an ein Tutorial:

Lifted Probabilistic Inference
in Relational Models

Guy Van den Broeck Dan Suciu
KU Leuven U. of Washington

SAT und #SAT

Gegenben eine KNF-Formel

- SAT: Finde erfüllende Belegung
  - NP-vollständig: SAT, graph coloring, Hamiltonian cycle, …
- #SAT: Zähle erfüllende Belegungen
  - #P-vollständig: #SAT, …
- Beispiel: $F = (x \vee y) \wedge (y \vee \neg z)$
  - 5 erfüllende Belegungen
  - (0,1,0), (0,1,1), (1,1,0), (1,1,1), (1, 0, 0)

Wiederholung: DPLL(KNF-Formel)

DPLL(F)

if F is empty **then** return 1

if F contains an empty clause **then** return 0

else choose a variable x to branch

return $\mathrm{DPLL}(F|_{x=1}) \vee \mathrm{DPLL}(F|_{x=0})$

#DPLL(F) // computes expected satisfying probability of F

if F is empty then **return** 1

if F contains an empty clause **then** return 0

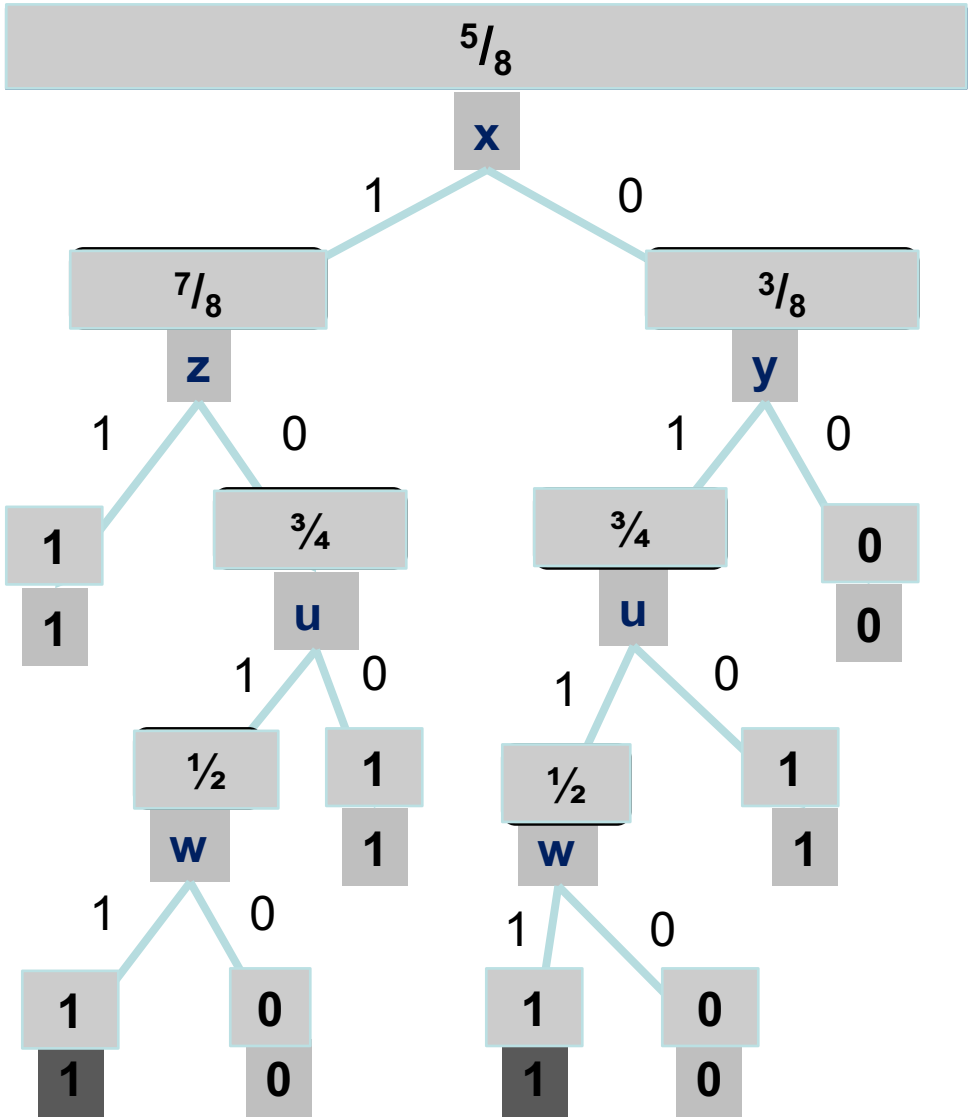
else choose a variable x to branch

return $0.5 * \#\mathrm{DPLL}(F|_{x=1}) + 0.5 * \#\mathrm{DPLL}(F|_{x=0})$

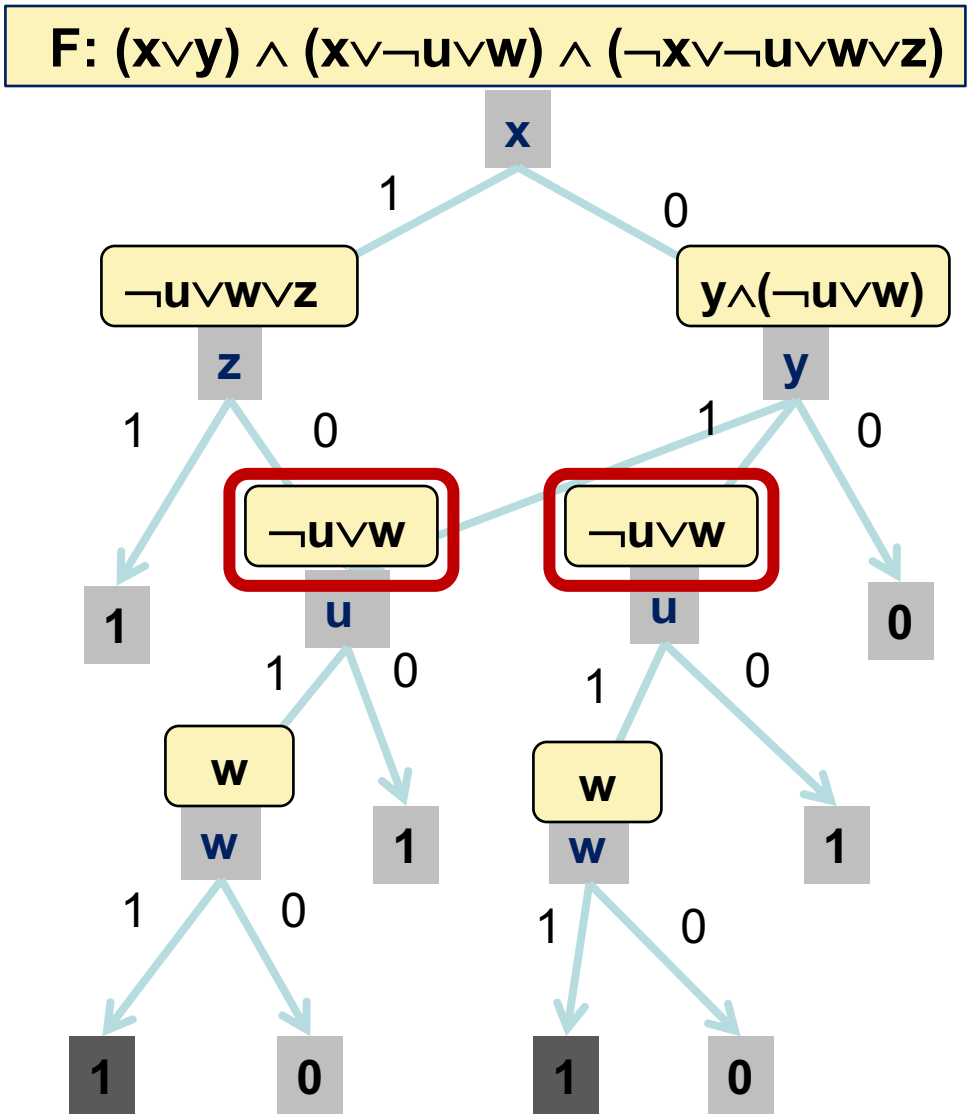
Davis, Martin; Putnam, Hilary. "A Computing Procedure for Quantification Theory". Journal of the ACM. 7 (3): 201–215. **1960**.

Davis, Martin; Logemann, George; Loveland, Donald. "A Machine Program for Theorem Proving". Communications of the ACM. 5 (7): 394–397. **1962**.

DPLL: Beispiel



Caching



Weighted Model Counting (WMC)

- Jedes Literal hat Gewicht $\in$ [0,1]
  - Gewicht eines Modells = Produkt der Literalgewichte
  - Gewicht einer Formel = Summe der Gewichte der Modelle

Basic Weighted Model Counting (BWMC)

BWMC(F)

if F is empty **then return** 1

if F contains an empty clause **then return** 0

else choose a variable x to branch

return (BWMC($F|_{x=1}$) * weight (x) +

BWMC($F|_{x=0}$) * weight (¬x))

Weighted Model Counting

- Modell = Erfüllende Belegung einer aussagenlogischen Formel Δ
- Zählen der Modelle: #SAT
- Weighted Model Counting (WMC)
  - Gewichte für Variablenzuweisung
  - Modellgewicht = Produkt der Variablengewichte w(·)

Δ = (Rain ⇒ Cloudy)

Rain

w(R)	w(¬R)
1	2

Cloudy

w(C)	w(¬C)
3	5

Rain	Cloudy	Model?	Weight
T	T	Yes	1 * 3 = 3
T	F	No	0
F	T	Yes	2 * 3 = 6
F	F	Yes	2 * 5 = 10
		+	+
		#SAT = 3	WMC = 19

Weighted First-Order Model Counting

Modell = Erfüllende Belegung einer aussagenlogischen Formel Δ

$\Delta = \forall d\ (\mathrm{Rain}(d) \Rightarrow \mathrm{Cloudy}(d))$

Days = {Monday **Tuesday**}

Rain

d	w(R(d))	w(¬R(d))
M	1	2
T	4	1

Cloudy

d	w(C(d))	w(¬C(d))
M	3	5
T	6	2

Rain(M)	Cloudy(M)	Rain(T)	Cloudy(T)	Model?	Weight
T	T	T	T	Yes	1 * 3 * 4 * 6 = 72
T	F	T	T	No	0
F	T	T	T	Yes	2 * 3 * 4 * 6 = 144
F	F	T	T	Yes	2 * 5 * 4 * 6 = 240
T	T	T	F	No	0
T	F	T	F	No	0
F	T	T	F	No	0
F	F	T	F	No	0
T	T	F	T	Yes	1 * 3 * 1 * 6 = 18
T	F	F	T	No	0
F	T	F	T	Yes	2 * 3 * 1 * 6 = 36
F	F	F	T	Yes	2 * 5 * 1 * 6 = 60
T	T	F	F	Yes	1 * 3 * 1 * 2 = 6
T	F	F	F	No	0
F	T	F	F	Yes	2 * 3 * 1 * 2 = 12
F	F	F	F	Yes	2 * 5 * 1 * 2 = 20
				+	+
				#SAT = 9	**WFOMC = 608**

Gogate, V., & Domingos, P. Probabilistic theorem proving. Proc. UAI, **2012**.

Van den Broeck, G., Taghipour, N., Meert, W., Davis, J., & De Raedt, L., Lifted probabilistic inference by first-order knowledge compilation. In Proc.IJCAI-11, pp. 2178-2185, **2011**.

Von Wahrscheinlichkeiten zu Gewichten

Friend

x	y	P
A	B	~~p_1~~
A	C	~~p_2~~
B	C	~~p_3~~

→

x	y	w(Friend(x,y))	w(¬Friend(x,y))
A	B	$w_1 = p_1$	$\underline{w}_1 = 1-p_1$
A	C	$w_2 = p_2$	$\underline{w}_2 = 1-p_2$
B	C	$w_3 = p_3$	$\underline{w}_3 = 1-p_3$
A	A	$w_4 = 0$	$\underline{w}_4 = 1$
A	C	$w_5 = 0$	$\underline{w}_5 = 1$
	…	…	

Also for missing tuples!

$w_1 w_2 w_3$

$\cancel{p_1 p_2 p_3}$

$\cancel{(1-p_1)p_2 p_3}$

$\underline{w}_1 w_2 w_3$

$\cancel{(1-p_1)(1-p_2)(1-p_3)}$

$\underline{w}_1 \underline{w}_2 \underline{w}_3$

Intensionale Anfrageevaluation

- Anfrage Q + ProbDB D

  => Herkunftsformel (*lineage expression)* F
  - Boolesche Variablen w_1, $\underline{w}_1$, ...
    korrespondierend zu Tupeln t1, t2, ...
  - Die Herkunftsformel F sagt, wann Q wahr ist
- Berechne P(F) mit DPLL-artigem System

P. Agrawal, O. Benjelloun, A. Das Sarma, C. Hayworth, S. Nabar, T. Sugihara, and J. Widom. Trio: A System for Data, Uncertainty, and Lineage. Proc. 32nd Intl. Conference on Very Large Data Bases, pages 1151-1154, **2006**

J. Widom. Trio: A System for Data, Uncertainty, and Lineage. In C. Aggarwal, editor, Managing and Mining Uncertain Data, Springer, **2009**.

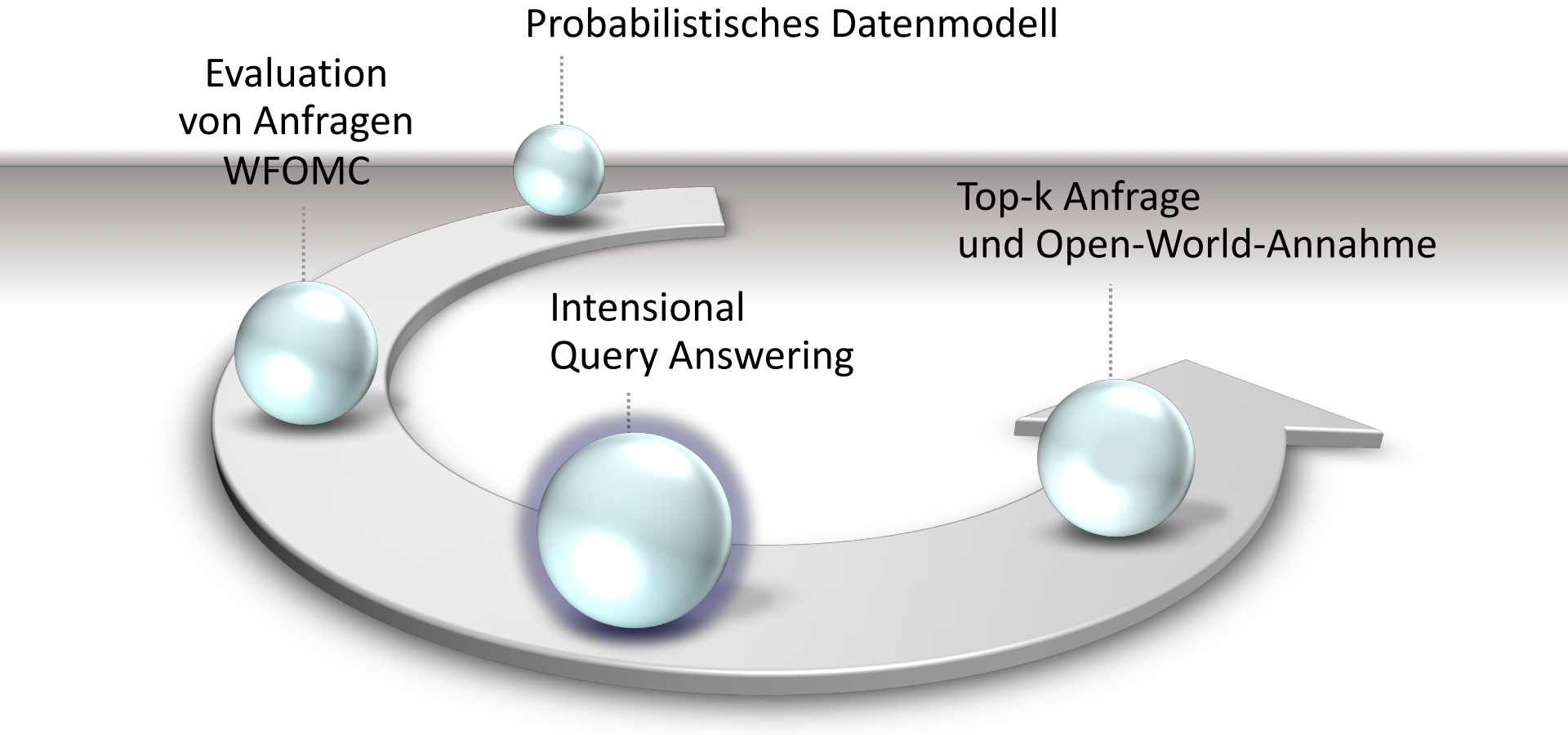
Diskussion

- Einfache Idee: Ersetze p, 1-p durch w, w
- Anfragebeantwortung durch WFOMC
- Für Wahrscheinlichkeitsraum:
  - Dividiere Weltgewicht durch Z = Summe aller Weltgewichte

- Warum Gewichte statt Wahrscheinlichkeiten?
- Verschiedene Formalismen zur Darstellung von Wahrscheinlichkeitsverteilungen

Non-Standard-Datenbanken

Probabilistische Datenbanken



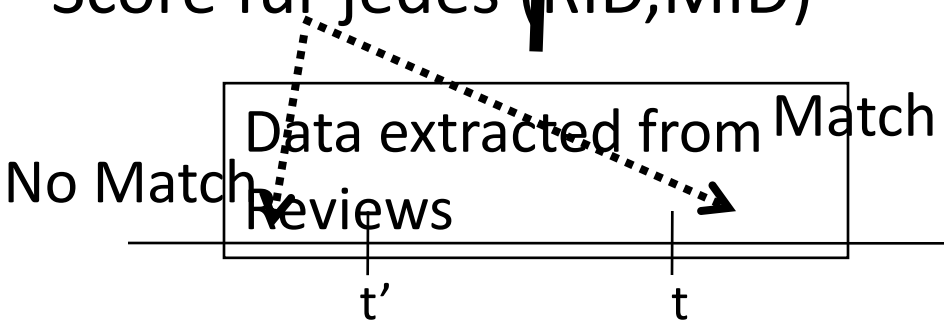
Unschärfe ist überall…

RID	Title
r124	12 Monkeys
r155	Twelve Monkeys
r175	2 Monkey
r194	Monk

MID	Title
m232	12 Monkeys
m143	Monkey Love

Fellegi-Sunter-Ansatz:
Score für jedes (RID,MID)

Clean IMDB Data



Output: (RID,MID) pairs

Our Approach: Convert scores to probabilities

Fellegi, Ivan; Sunter, Alan. "A Theory for Record Linkage" Journal of the American Statistical Association. 64 (328): pp. 1183–1210, **1969**.

Unschärfe ist überall…

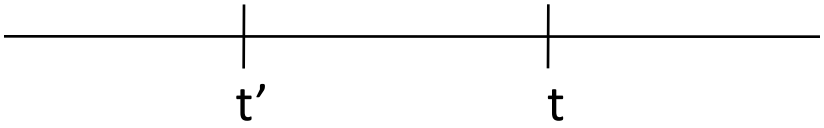
RID	Title
r124	12 Monkeys
r155	Twelve Monkeys
r175	**2 Monkey**
r194	Monk

MID	Title
m232	**12 Monkeys**
m143	**Monkey Love**

Fellegi-Sunter-Ansatz:

Score für jedes (RID,MID)

No Match Match



RID	MID	Prob
r175	m232	0.8
r175	m143	0.2

Fellegi, Ivan; Sunter, Alan. "A Theory for Record Linkage" Journal of the American Statistical Association. 64 (328): pp. 1183–1210, **1969**.

Anfragebeantwortung mit Herkunftsformel

- Intensionale Anfrage-beantwortung [FR97]
- Assoziiere mit jedem Tupel ein Ereignis
- Wahrscheinlichkeit, dass Ereignis eintritt / erfüllt ist = Anfragewert

RID	MID	Prob	
r175	m232	0.8	e_1
r175	m143	0.2	e_2

Anfrageverarbeitung generiert Ereignisausdruck (Lineage-Formel)

Kreuzprodukt	$\times((t_1,[e_1]),(t_2,[e_2]))$	$=((t_1,t_2),[e_1 \wedge e_2])$
Selektion	$\sigma_c((t,[e]))$	$= \begin{cases} (t,[e]) & c(t) = \text{true} \\ \perp & c(t) = \text{false} \end{cases}$
Projektion	$\Pi_A((t_1,[e_1]),\ldots,(t_n,[e_n]))$	$=(t_1[A],[e_1 \vee \cdots \vee e_n])$

N. Fuhr; T. Rölleke, A Probabilistic Relational Algebra for the Integration of Information Retrieval and Database Systems. ACM Transactions on Information Systems 14(1), **1997**.

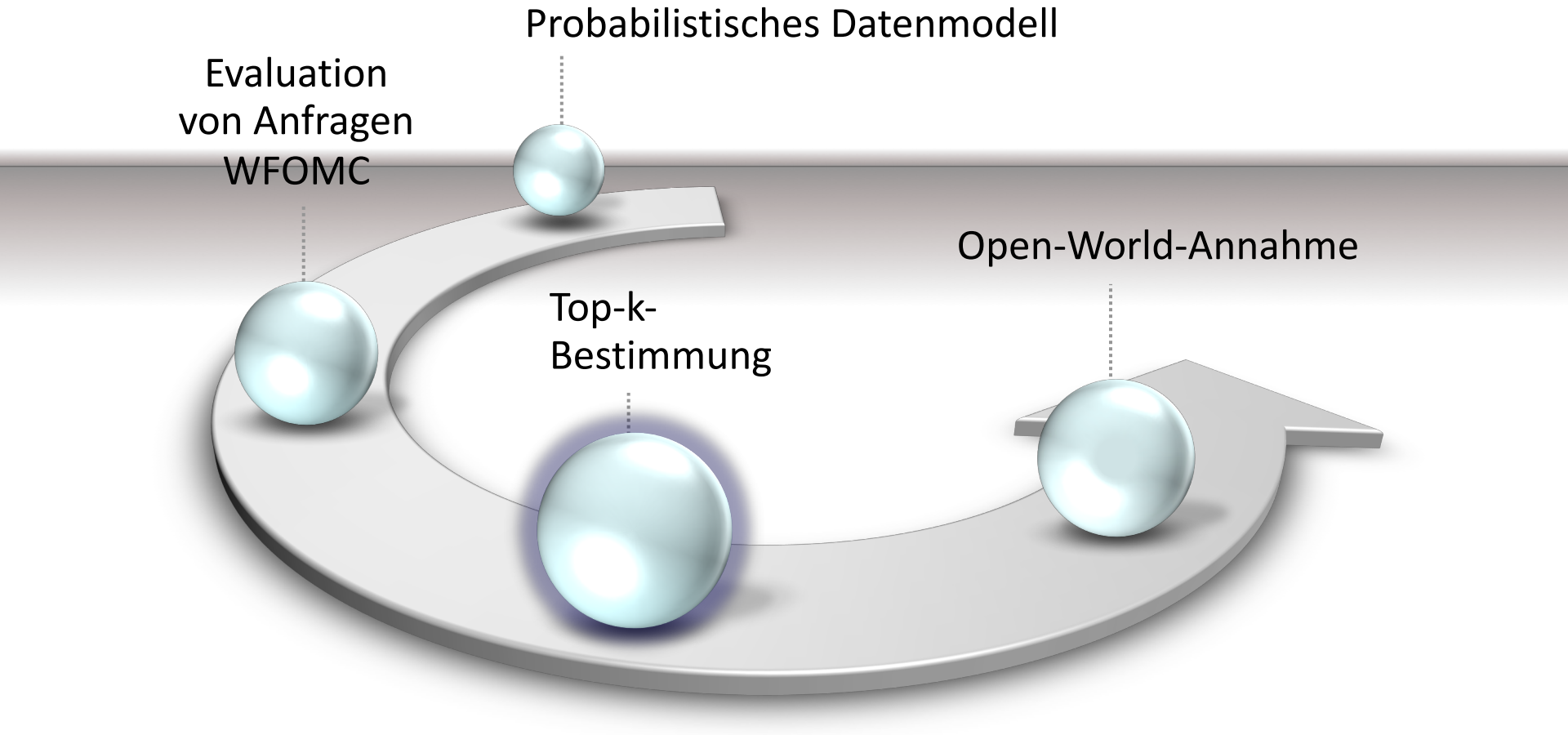
Intensionale Anfrageevaluation

- Boolesche Anfrage Q + ProbDB D
  => Herkunftsformel (*lineage expression)* F
  - Boolesche Variablen e_1, $\underline{e}_1$, ...
    und damit Gewichte w_1, $\underline{w}_1$, ...
    korrespondierend zu Tupeln t_1, t_2, ...
    der extensionalen DB
  - Die Herkunftsformel F sagt, wann Q wahr ist
- Berechne P(F) mit #DPLL-artigem System und WFOMC

- Für Anfragen mit Parametern ist das für jedes Ergebnistupel t zu berechnen.

Non-Standard-Datenbanken

Probabilistische Datenbanken



Top-k-Anfragebeantwortung

Gegeben: $G = \{t_1, \ldots, t_n\}$ eine Menge von n Objekten mit unbekannten Wahrscheinlichkeiten $p_1, \ldots, p_n$ und eine Zahl $k \leq n$.

Ziel: Finde Menge von k Objekten mit höchsten Wahrscheinlichkeiten, genannt Top-k-Teilmenge von G

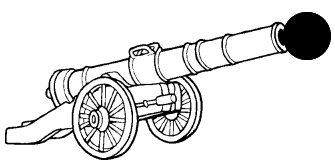
Lösungsidee: Verwende parallele Monte-Carlo-Simulationen, eine für jede Kandidatenantwort (Lineage-Formel in DNF), und approximiere Wahrscheinlichkeiten nur soweit, wie es nötig ist, um die k besten Antworten zu finden

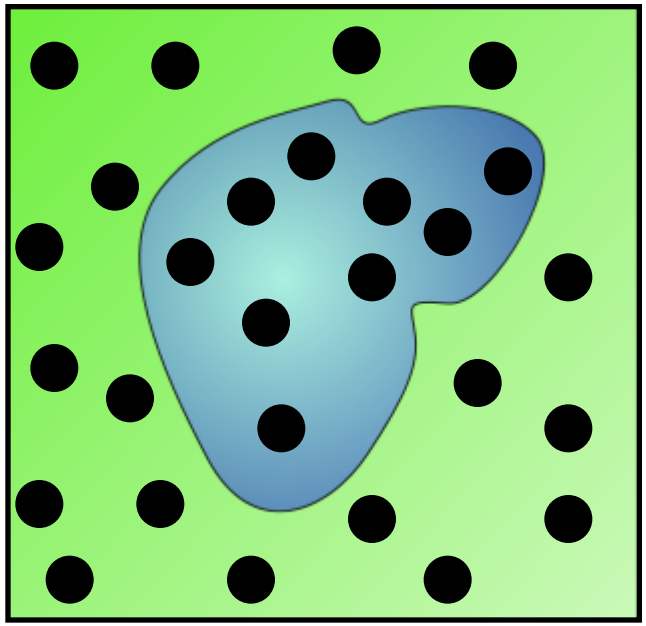
Monte-Carlo-Simulation: Einführung

- Gegeben: Gesamtfläche eines Landstücks (Quadrat)
- Wie kann die Fläche des darauf liegenden Sees bestimmt werden?

$\text{Fläche}_{\text{Land}}$ = 1000 m² (Quadrat)
X = Anzahl Kanonenschüsse
N = Einschläge auf grünem Land

$$\frac{\text{Fläche}_{\text{Land}}}{\text{Fläche}_{\text{See}}} = \frac{X}{X-N} \quad \Rightarrow \quad \text{Fläche}_{\text{See}} = \frac{(X-N)\times \text{Fläche}_{\text{Land}}}{X}$$





$\text{Fläche}_{\text{See}} = 1000$

$\text{Fläche}_{\text{See}} = 500$

$\text{Fläche}_{\text{See}} = 333.\overline{3}$

...

$\text{Fläche}_{\text{See}} = 375$

Monte Carlo Simulation: (ε, δ)-Approximation

Function MS-Naiv(G)

Wähle N mal zufällig eine mögliche Welt.

Berechne Wahrheitswert von DNF-Formel G.

Wahrscheinlichkeit p=P(G) approximiert durch Frequenz $\tilde{p}$, mit der G wahr wird

Theorem: If $N \geq (1/\Pr(G)) \times (4 \ln(2/\delta)/\varepsilon^2)$
then: $\Pr[\ |P/\Pr(E) - 1| > \varepsilon\ ] < \delta$

N may be very big for small Pr(E)

Function MS-Karp-Luby(G)

Fix an order on the disjuncts: $\{t_1, t_2, \ldots, t_m\} = G$

$C := 0$

repeat // importance sampling

Choose a random disjunct $t_i \in G$

Choose a random truth assignment s.t. $t_i = true$

if forall $j < i$ *it holds that* $t_j = false$ then $C := C + 1$

until *N times*

return C/N

Works for any G (but <u>not</u> in PTIME)

This is better!

$\tilde{p}$:= Karp-Luby(G)

$$\varepsilon = \sqrt{4m \log(2/\delta)/N} \quad a^N = \tilde{p} - \varepsilon \quad b^N = \tilde{p} + \varepsilon$$

Works in PTIME for G in DNF.

Richard M. Karp, Michael Luby, Monte-Carlo Algorithms for Enumeration and Reliability Problems. FOCS: 56-64, **1983**.

R. M. Karp, M. Luby, and N. Madras. Monte-Carlo approximation algorithms for enumeration problems. J. Algorithms 10, 3 (September 1989), 429-448. **1989**

Paul Dagum, Richard M. Karp, Michael Luby, Sheldon M. Ross: An Optimal Algorithm for Monte Carlo Estimation. SIAM J. Comput. 29(5): 1484-1496, **2000**.

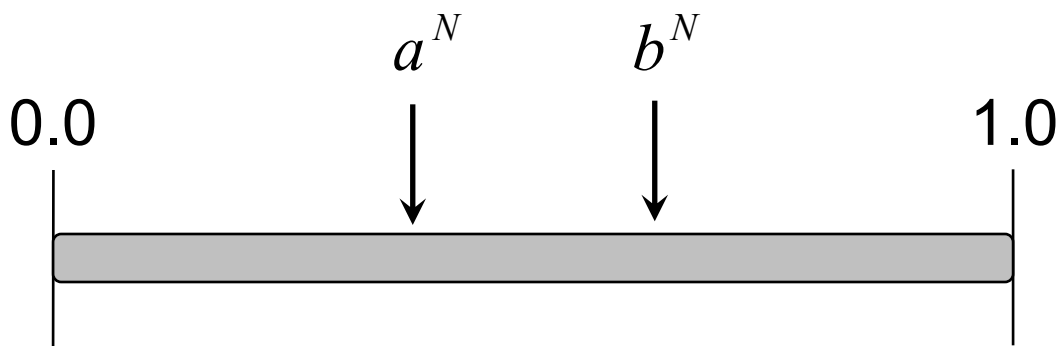
Karp-Luby-Madras

Nach N Simulationen garantiert:

$$\mathbf{P}(p \in [a^N, b^N]) \quad > \quad 1 - \delta$$

Konfidenzintervall

Die Sicherheit bezüglich der Wahrscheinlichkeit



Anfragebeantwortung

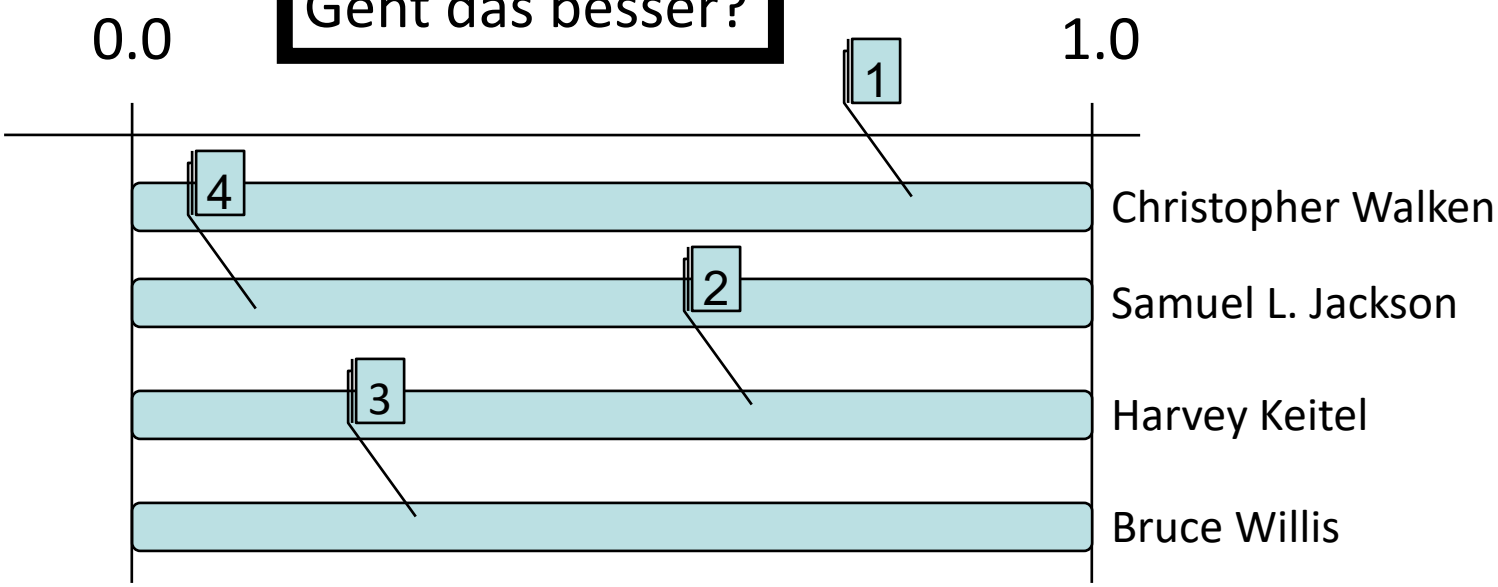
Anfrage: Finde Top-k Regisseure von guten Filmen (Score ≥ 4)
NB: Ranking nach P-Werten

Verfahren: Simuliere für jeden Kandidaten mit Karp-Luby bis ε klein

MID	Director
m232	Christopher Walken
…	…

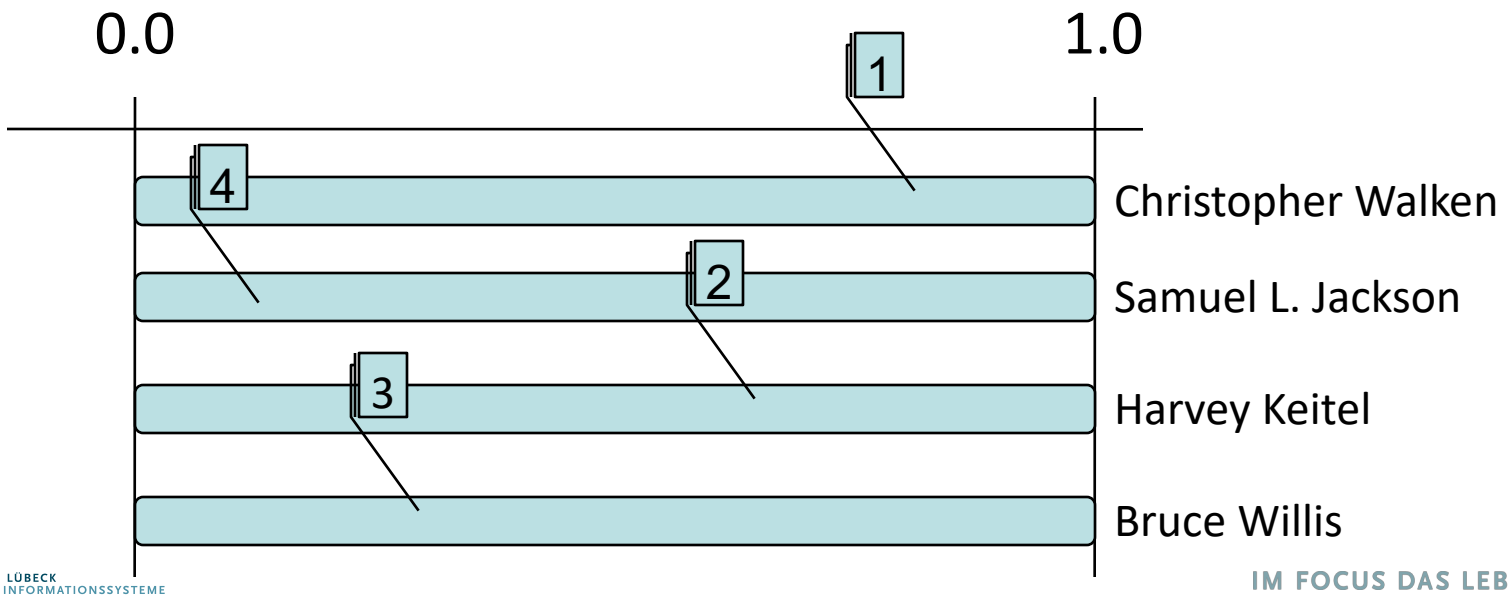
RID	Score
r157	4
…	…

Geht das besser?



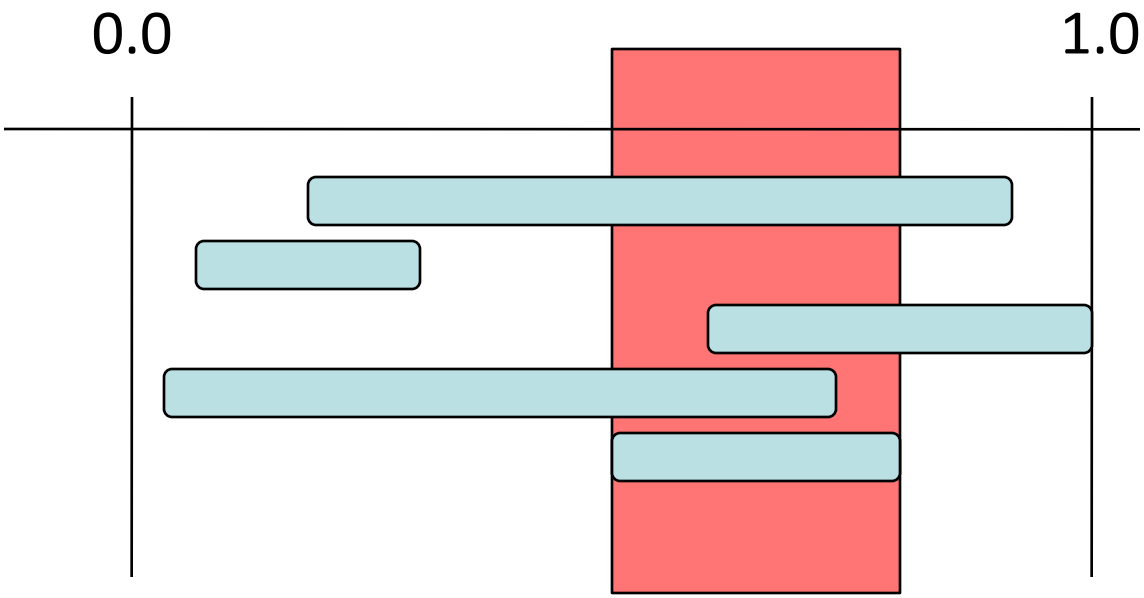
Besseres Verfahren: Multisimulation

- Trenne Top-K mit wenigen Simulationen
  - Betrachte Intervalle im Top-k-Bereich
  - Am Ende: Intervalle verschränkt
- Vergleich mit imaginärem Verfahren OPT
  - *OPT* “Weiß,” welche Intervall betrachtet werden müssen



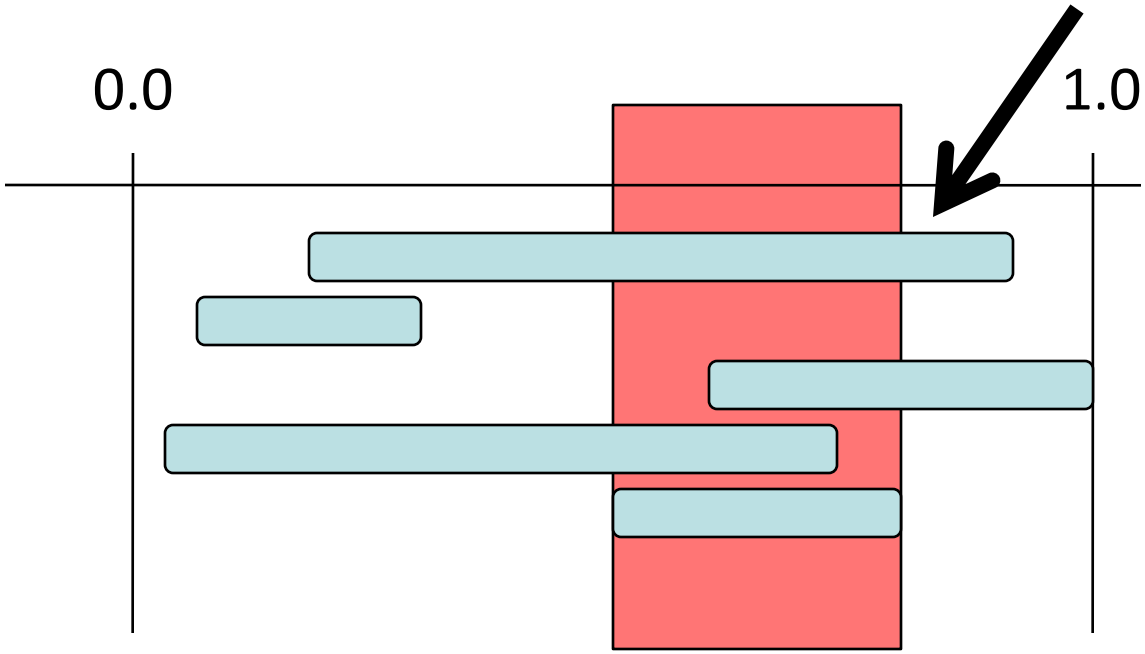
Karp-Luby-Madras im Einsatz: Die kritische Region

- Die kritische Region ist das Intervall
  - (k-höchste Min, k+1-höchste Max)
  - Für k = 2



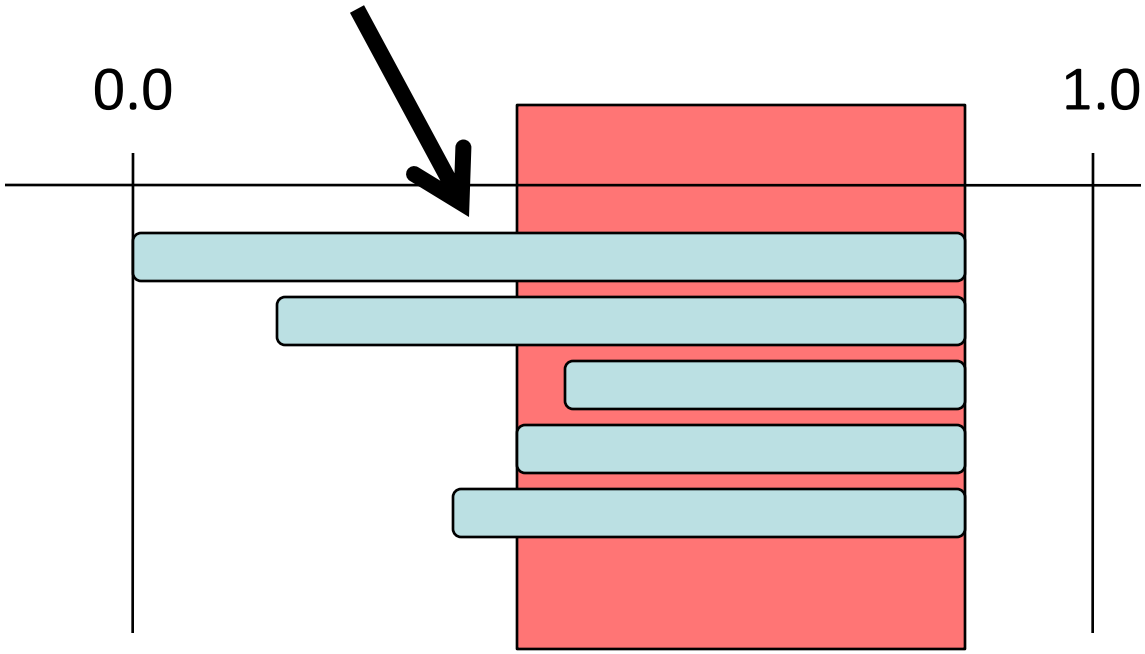
Drei einfache Regeln: Regel 1

- Wähle “Double Crosser”
  - *OPT* muss dieses Intervall auch wählen



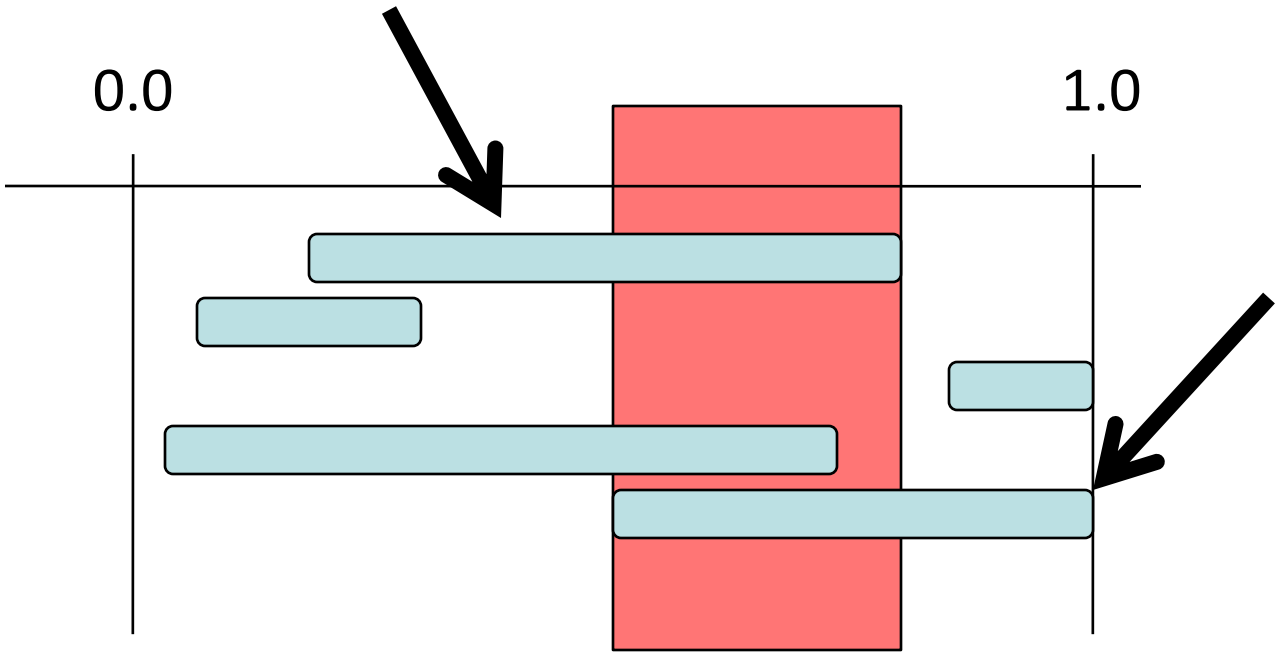
Drei einfache Regeln: Regel 2

- Nur noch "Lower/Upper Crosser"? dann wähle maximales Intervall
  - *OPT* muss das auch machen



Drei einfache Regeln: Regel 3

- Wähle Upper- und Lower-Crosser
  - *OPT* könnte nur ein Intervall wählen



Multisimulation (MS)

function MS_TopK($\{G_1, \ldots, G_n\}$, k):

  $[a_1, b_1] := \ldots := [a_n, b_n] = [0, 1]$

repeat

  $(c, d) := (topk(a_1, \ldots, a_n), topk+1(b_1, \ldots, b_n))$

  $T := \{G_i \mid d \leq a_i\}$

  $B := \{G_i \mid b_i \leq c\}$

  Case 1: choose a double crosser to simulate(T, B)

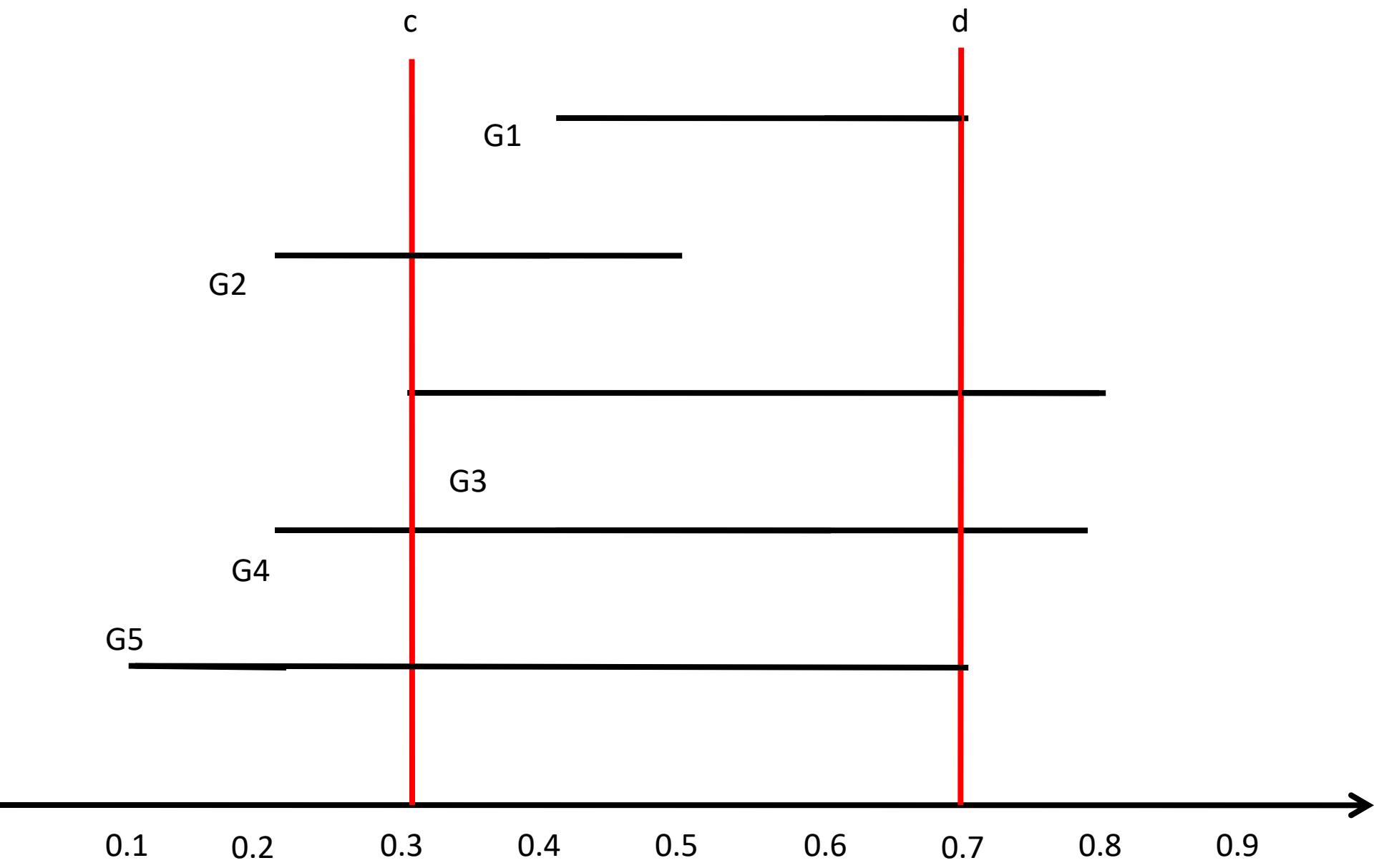
  Case 2: choose upper and lower crosser to simulate(T, B)

  Case 3: choose a maximal crosser to simulate(T, B)

until $c > d$

return T

Example : Let us select Top 2



Multisimulation ist 2-approximierend

- Theorem [DS07]: Multisimulation führt höchstens zweimal soviele Simulationen aus wie OPT
  - Und: kein deterministischer Algorithmus kann auf beliebigen Probleminstanzen besser arbeiten
- Varianten
  - Top-k-Menge (gezeigt)
  - Anytime (produziere von 1 bis k)
  - Rang (produziere top-k nach Rang sortiert)
  - Alle (alle Intervalle nach Rang sortiert)

Nilesh Dalvi and Dan Suciu. 2007. Efficient query evaluation on probabilistic databases. *The VLDB Journal* 16, 4, 523-544, **2007**.

Experimente: Unsichere Tupel

Table	# Tuples
StringMatch	339k
ActorMatch	6,758k
DirectorMatch	18k

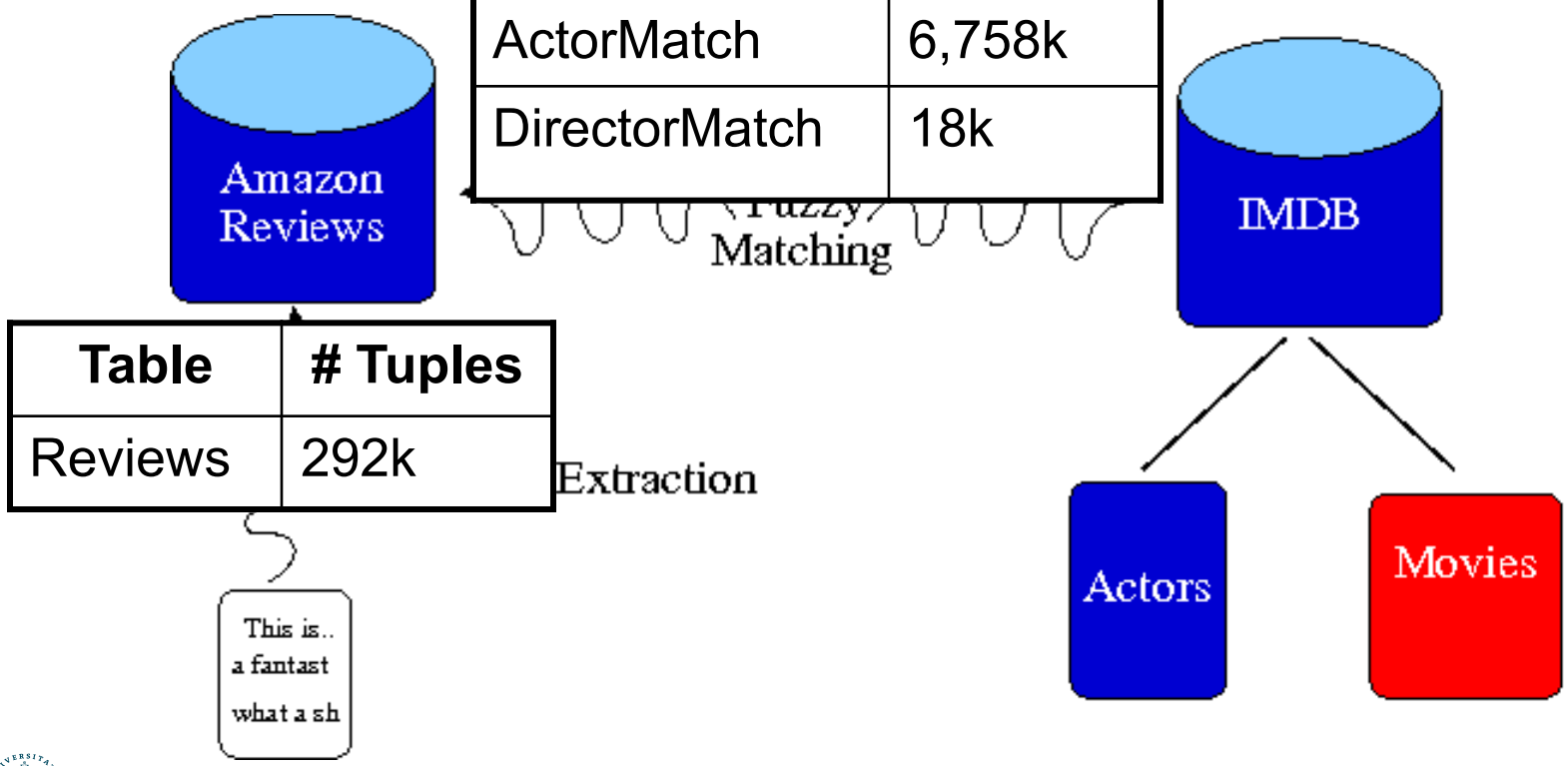
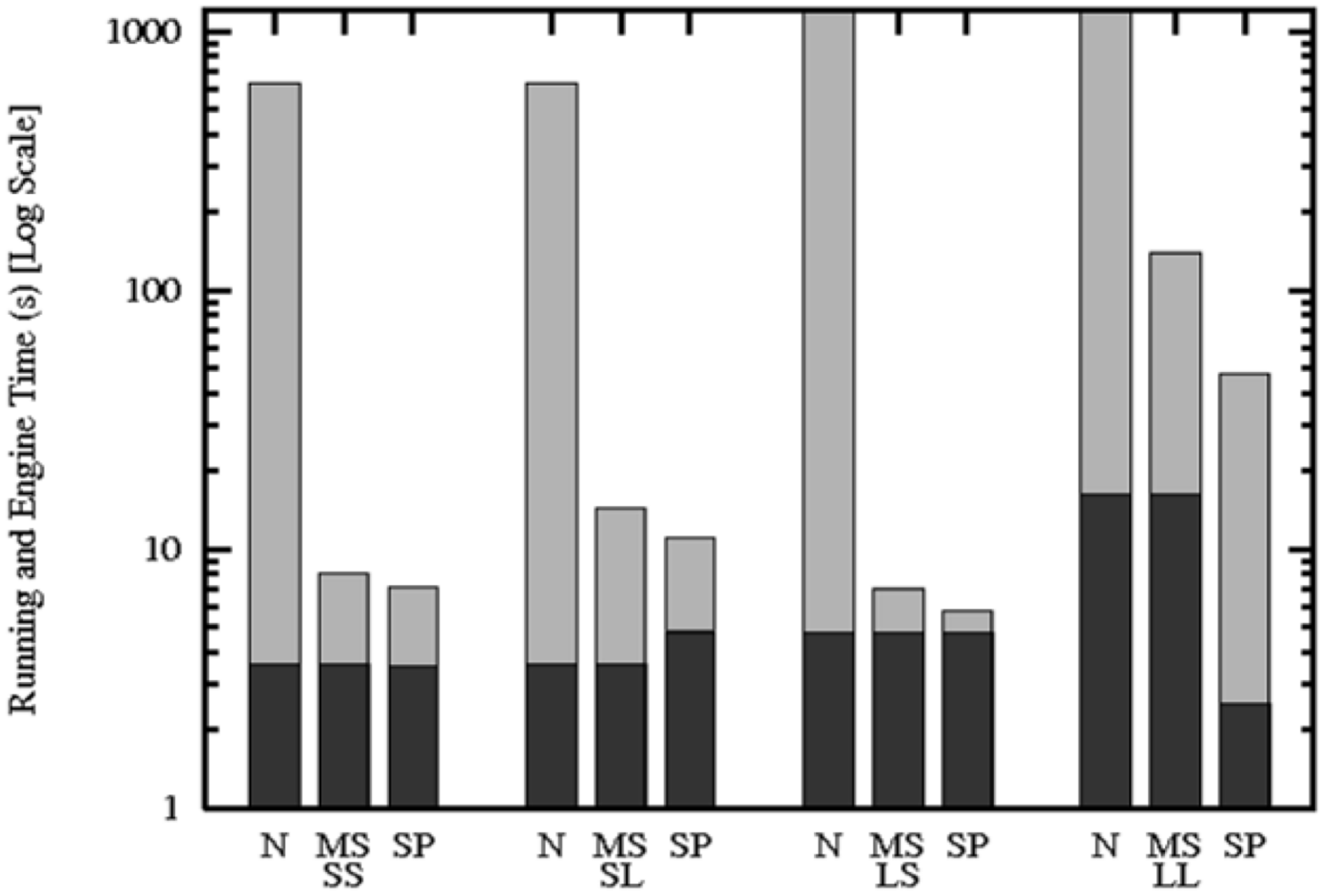
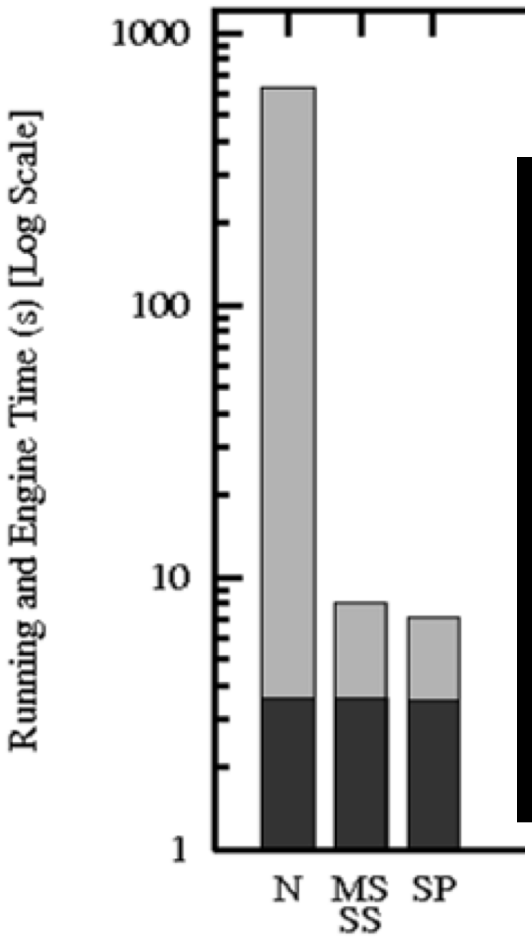


Table	# Tuples
Reviews	292k

Laufzeiten [**N(aiv), MS, and S(afe) P(lan)]**



Laufzeiten



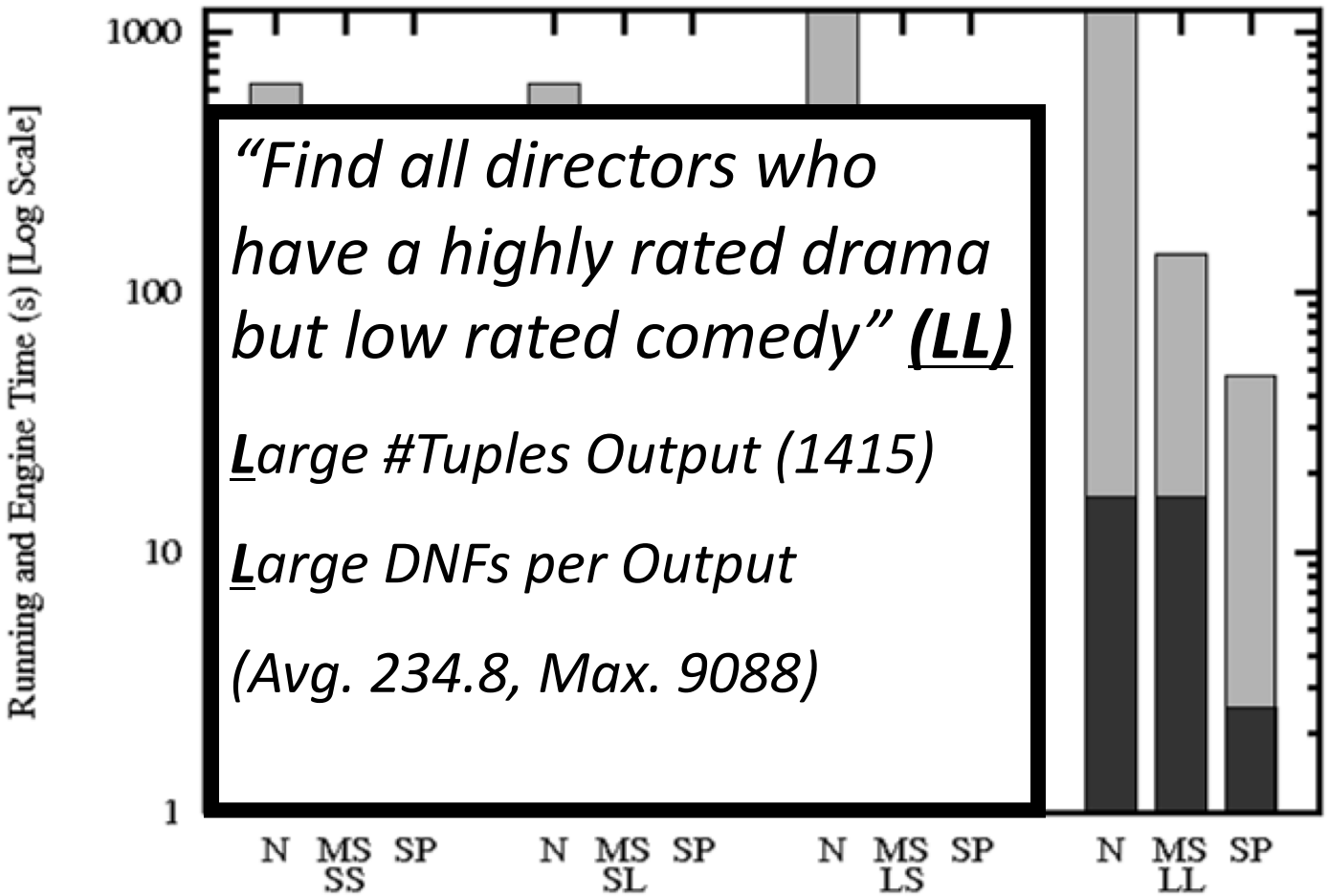
"Find all years in which Anthony Hopkins was in a highly rated movie" ***(SS)***

S*mall Number of Tuples Output (33)*

S*mall DNFs per Output*

(Avg. 20.4, Max 63)

Laufzeiten



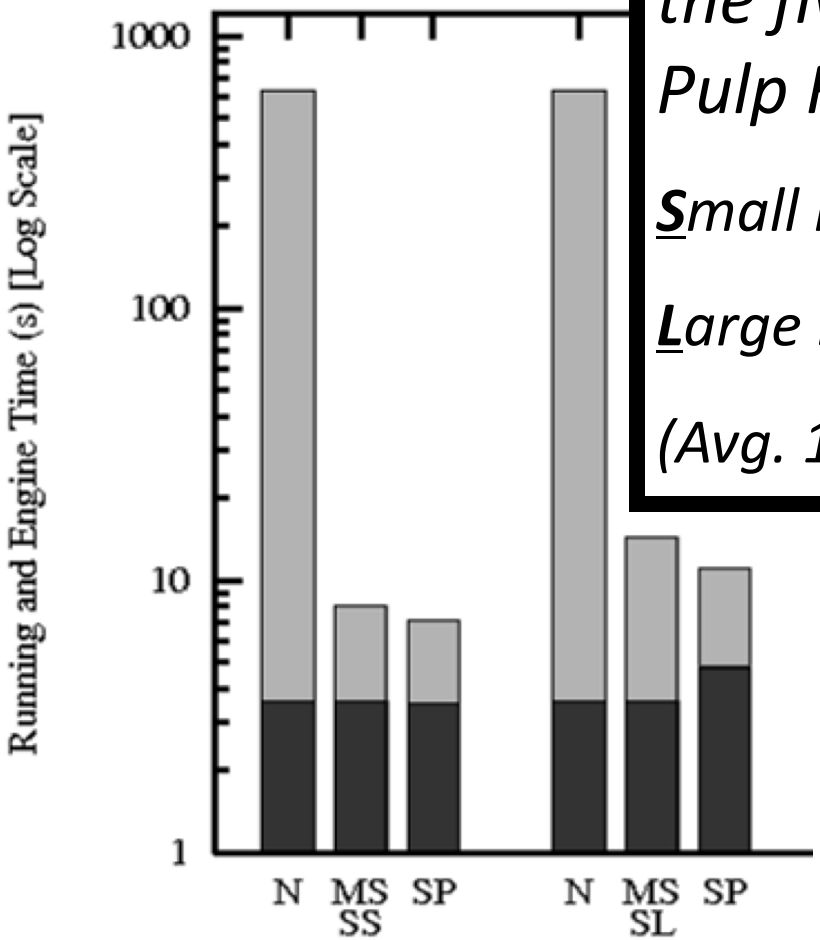
"Find all directors who have a highly rated drama but low rated comedy" ***(LL)***

L*arge #Tuples Output (1415)*

L*arge DNFs per Output*

(Avg. 234.8, Max. 9088)

Laufzeiten



"Find all actors in Pulp Fiction who appeared in two very bad movies in the five years before appearing in Pulp Fiction" ***(SL)***

S*mall Number of Tuples Output (33)*

L*arge DNFs per Output*

(Avg. 117.7,Max 685)

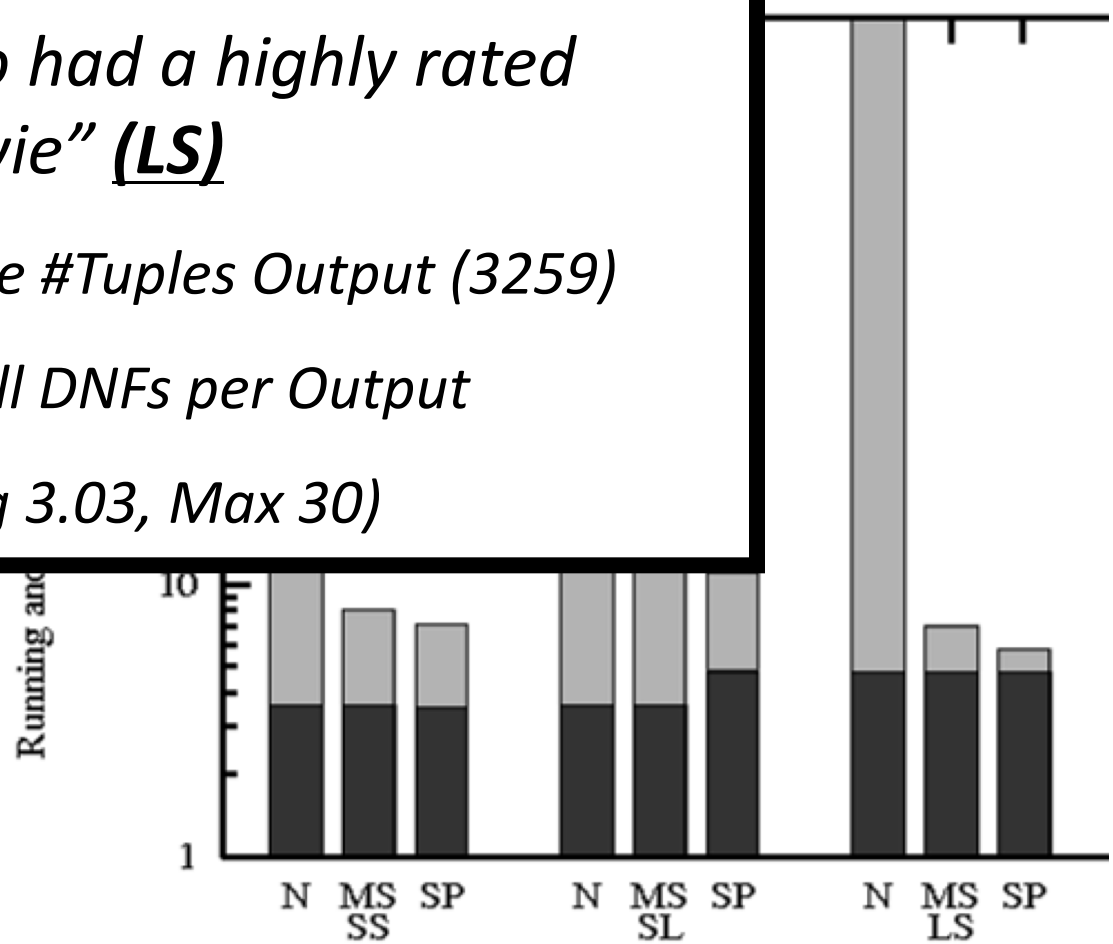
Laufzeiten

"Find all directors in the 80s who had a highly rated movie" ***(LS)***

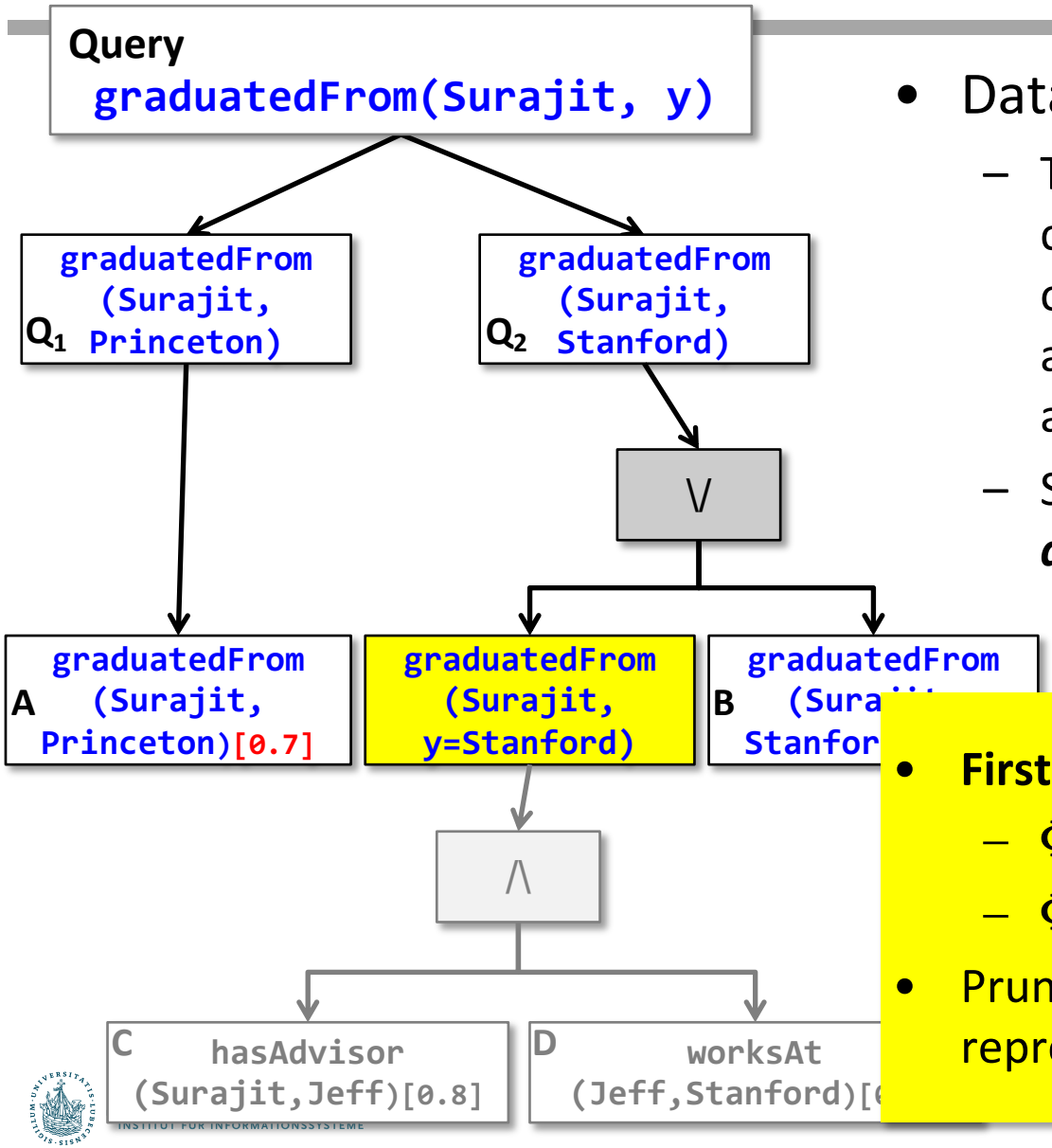
L*arge #Tuples Output (3259)*

S*mall DNFs per Output*

(Avg 3.03, Max 30)



Top-*k* Ranking by Marginal Probabilities: First-Order Lineage

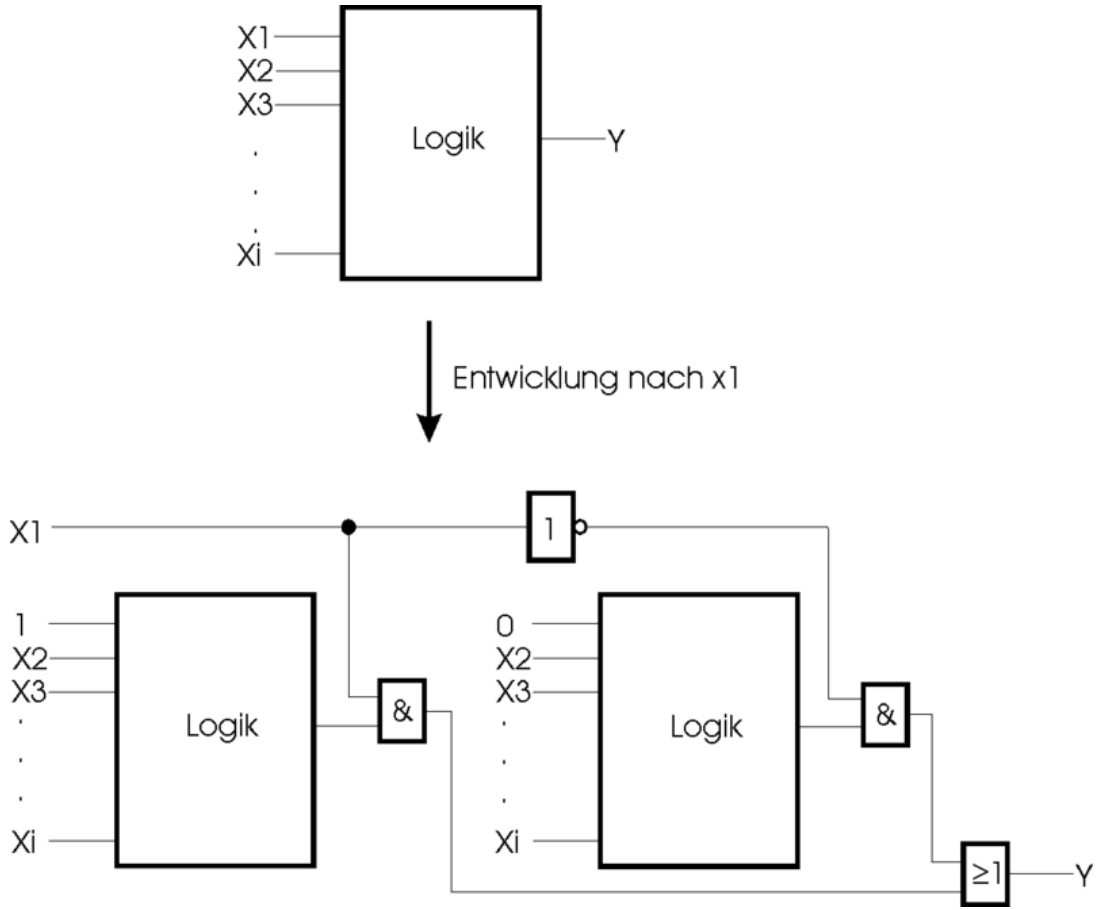


- Datalog/SLD resolution
  - Top-down grounding allows us to compute ***lower*** and ***upper bounds*** on the marginal probabilities of answer candidates before rules are fully grounded.
  - Subgoals may ***represent sets of answer candidates***.

- **First-order lineage** formulas:
  - $\Phi(Q_1) = A$
  - $\Phi(Q_2) = B \vee \exists y\ \mathbf{gradFrom(Surajit,y)}$
- Prune entire set of answer candidates represented by Φ.

Shannon Expansion

$$F(x_1, x_2, \ldots, x_n) = x_i F(x_1, \ldots, x_{i-1}, 1, x_{i+1}, \ldots, x_n) \vee \overline{x}_i F(x_1, \ldots, x_{i-1}, 0, x_{i+1}, \ldots, x_n)$$



[Wikipedia]

Approximation: Hybride Ansätze

- Gegeben: Lineage-Formel
- Gesucht: Erwartete gewichtete Erfüllbarkeit
- Ansatz mit #DPLL oder WFOMC kann zu langsam sein
- Safe-Plan- und Shannon Expansions-Regeln auf (FO-)Lineage-Formel bis zu einem gewissen Level ausführen
- Dann Karp-Luby Monte-Carlo-Simulation zur Unsicherheitsverringerung bzgl. der Lineage-Formel
  - ggf. Mittelwert aus Intervall für konkreten P-Wert nehmen
  - Gute Approximation (Sättigung)
    bei geringer Shannon-Expansionstiefe
  - Vertretbare Rechenzeit
  - In Bachelor-Arbeit von Simon Paasche in Healthcare-Szenario evaluiert

Maximilian Dylla, Iris Miliaraki, Martin Theobald:
A Temporal-Probabilistic Database Model for Information Extraction.
PVLDB 6(14): 1810-1821, **2013**

Literatur

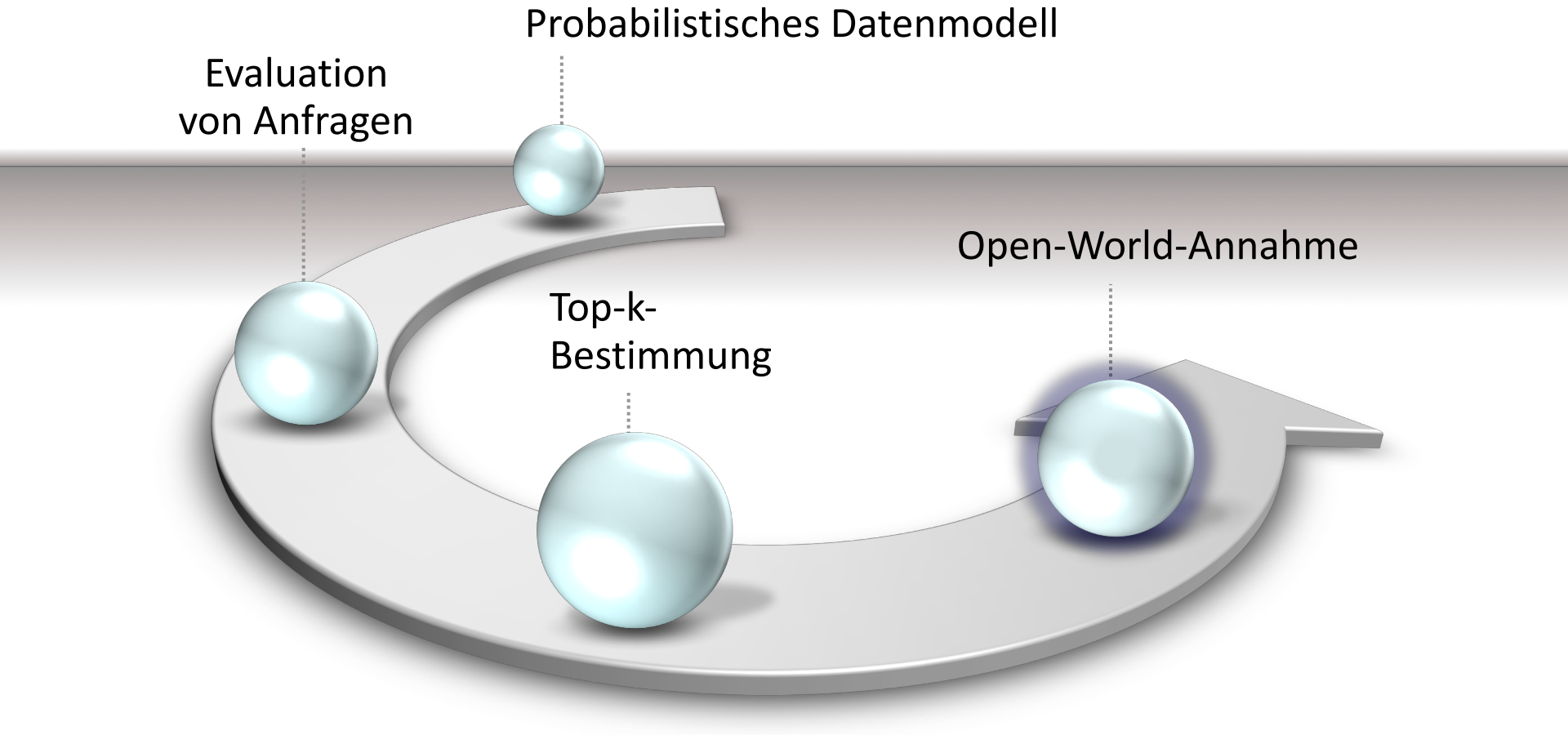
Ronald Fagin, Amnon Lotem, and Moni Naor. Optimal aggregation algorithms for middleware. In Proceedings of the 20th ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems (PODS '01). ACM, New York, NY, USA, 102-113., **2001**

Balke, W. - T., U. Güntzer, and W. Kießling, "On Real-time Top k Querying for Mobile Services", 10th International Conference on Cooperative Information Systems (CoopIS 2002), Irvine, CA, USA, Springer, **2002**

Maximilian Dylla, Iris Miliaraki, Martin Theobald:
A Temporal-Probabilistic Database Model for Information Extraction.
PVLDB 6(14): 1810-1821, **2013**

Non-Standard-Datenbanken

Probabilistische Datenbanken



Danksagung

Die nachfolgenden Präsentationen sind aus einem Vortrag "Open World Probabilistic Databases" von Ismail Ilkan Ceylan, Adnan Darwiche und Guy Van den Broeck übernommen und wurden ggf. modifiziert.

İsmail İlkan Ceylan, Adnan Darwiche, and Guy Van den Broeck. Open-world probabilistic databases. In *Proc. Knowledge Representation and Reasoning* (KR'16), pp. 339-348, **2016**.

CWA vs OWA

Ein Flug taucht nicht in Flüge-Datenbank auf

Der Flug findet nie statt!

In einer Patientenakte ist eine Penicillin-Allergie nicht erwähnt

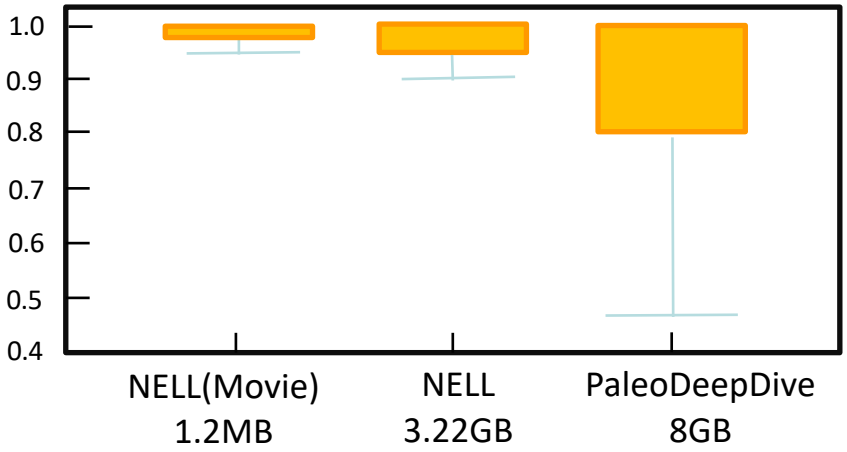
Der Patient leidet nicht an einer Penicillin-Allergie !

Einführung

- Unter der Annahme der geschlossenen Welt (Closed-World Assumption, CWA) wird als falsch angenommen, was nicht als wahr beweisbar ist
- Unter der Annahme der offenen Welt (Open-World Assumption, OWA) wird das, was nicht beweisbar ist, als unbekannt angenommen
- Das klassische PDB-Modell verwendet CWA
  - Fakten nicht in der DB haben Wahrscheinlichkeit 0
- Offene PDBs:
  - Fakten nicht in der DB haben "unbekannte" Wahrscheinlichkeit

Klasssische Anwendung

- Faktenextraktion aus Texten (DeepDive, Nell, Yago)
- Fakten beschrieben mit Sicherheitswerten
  - Deutung: Wie wahrscheinlich ist es, dass Faktum wahr?
- Verteilung der Sicherheitswerte:



- Vervollständigung? Unmöglich!

Probleme bei CWA in PDBs: Beispiele

Couple		P
arquette	cox	0.6
pitt	jolie	0.8
thornton	jolie	0.6
pitt	aniston	0.9
kunis	kutcher	0.7

Inmovie		P
w.Smith	ali	0.9
w.Smith	sharktable	0.8
j.james	ali	0.6
arquette	scream	0.7
pitt	mr ms smith	0.5
jolie	mr ms smith	0.7
jolie	sharktable	0.9

$$Q_1(x, y) = \text{Inmovie}(x, z), \text{Inmovie}(y, z), \text{Couple}(x, y)$$

$$Q_2 = \text{Inmovie}(x, z), \text{Inmovie}(y, z), \text{Couple}(x, y)$$

$Q_1(\text{pitt}, \text{jolie})$

Q_2

Erwartet:

$P\ (Q_1(\text{pitt}, \text{jolie}))\ <\ P(Q_2)$

Gefunden:

$P\ (Q_1(\text{pitt}, \text{jolie}))\ =\ P(Q_2) = 0.28$

Probleme bei CWA in PDBs: Beispiele

Couple		P
arquette	cox	0.6
pitt	jolie	0.8
thornton	jolie	0.6
pitt	aniston	0.9
kunis	kutcher	0.7

Inmovie		P
w.Smith	ali	0.9
w.Smith	sharktable	0.8
j.james	ali	0.6
arquette	scream	0.7
pitt	mr ms smith	0.5
jolie	mr ms smith	0.7
jolie	sharktable	0.9

$$Q_1(x, y) = \text{Inmovie}(x, z), \text{Inmovie}(y, z), \text{Couple}(x, y)$$

$Q1(\text{w. smith}, \text{j. james})$

$Q1(\text{thornton}, \text{aniston})$

Erwartet:

$P\ (Q_1(\text{w. smith}, \text{j. james}))\ >$

$P(Q1(\text{thornton}, \text{aniston}))$

Gefunden:

$P\ (Q_1(\text{w. smith}, \text{j. james}))\ =$

$P(Q1(\text{thornton}, \text{aniston})) = 0$

Probleme bei CWA in PDBs: Beispiele

Couple		P
arquette	cox	0.6
pitt	jolie	0.8
thornton	jolie	0.6
pitt	aniston	0.9
kunis	kutcher	0.7

Inmovie		P
w.Smith	ali	0.9
w.Smith	sharktable	0.8
j.james	ali	0.6
arquette	scream	0.7
pitt	mr ms smith	0.5
jolie	mr ms smith	0.7
jolie	sharktable	0.9

$$\boxed{Q_1(x, y) = \text{Inmovie}(x, z), \text{Inmovie}(y, z), \text{Couple}(x, y)}$$

$Q_1(\text{w. smith}, \text{j. james})$

$\text{Inmovie}(x, y) \wedge \neg\, \text{Inmovie}(x, y)$

$\text{Inmovie}(x, y) \wedge \neg \text{Inmovie}(x, y)$ ist nicht erfüllbar, wird aber gleich bewertet wie $Q_1(\text{w. smith}, \text{j. james})$

OpenPDBs

Inmovie		P
w.Smith	ali	0.9
w.Smith	sharktable	0.8
j.James	ali	0.6
arquette	scream	0.7
pitt	mr ms smith	0.5
jolie	mr ms smith	0.7
jolie	sharktable	0.9

Couple		P
arquette	cox	0.6
pitt	jolie	0.8
thornton	jolie	0.6
pitt	aniston	0.9
kunis	kutcher	0.7

Domain D

cruise
depp
hayek
pitt
j.James
.
.
Mission Impossible
Troy
Kingdom of Heaven

$\lambda \in [0, 1]$

<u>Offene Tupel für λ = 0.3:</u>
(Inmovie(pitt, Troy), 0.3)
(Inmovie(hayek, Mission Impossible), 0.15)

OpenPDBs

- Eine offene PDB P_λ ist ein Paar $(\mathcal{P}, \lambda)$,
  wobei $\mathcal{P}$ eine probabilistische DB ist und $\lambda \in [0, 1]$
- Für jedes Tupel nicht in $\mathcal{P}$ fügen wir ein Tupel $\langle t : p \rangle$
  hinzu für irgendein $p \in [0, \lambda]$

- P_λ induziert eine Menge von Wahrscheinlichkeitsverteilungen K_{P_λ}
  - Intervall-basierte Wahrscheinlichkeitsangaben für
    offene Tupel

OpenPDBs

- Das Wahrscheinlichkeitsintervall

  einer Booleschen Anfrage Q an P_λ ist

$$K_{P\lambda}(Q) = [\underline{P}_{P\lambda}(Q), \overline{P}_{P\lambda}(Q)]$$

wobei:

$$\underline{P}_{P\lambda}(Q) = \underset{P \in K_{P\lambda}}{\mathbf{min}}\ P(Q) \quad , \quad \overline{P}_{P\lambda}(Q) = \underset{P \in K_{P\lambda}}{\mathbf{max}}\ P(Q)$$

Beispiele - CWA vs OWA

$$Q_3 = \mathrm{Ac(patt), Workedfor(patt, hwicke), Di(hwicke)}$$

$$Q_4 = \mathrm{Ac(patt), Workedfor(patt, x), Di(x)}$$

Erwartet:

$$\bar{\mathrm{P}}\,(Q_4) > \bar{\mathrm{P}}(Q_3)$$

NELL Datenbank:

Gefunden:

$$\bar{\mathrm{P}}\,(Q_4) = \bar{\mathrm{P}}(Q_3) = 0$$

In einer offenen Welt:

Gefunden: für $\lambda = 0.3$

$$\bar{\mathrm{P}}\,(Q_4) = 0.82\,, \quad \bar{\mathrm{P}}(Q_3) = 0.51$$

Ac stands for actor, patt for Pattinson, Di for director, hwicke for Hardwicke, Mov for movie, and trainsp for Trainspotting

Beispiele - CWA vs OWA

$$Q_5 = Ac(x), Inmovie(x, trainsp), Mov(trainsp), \neg Di(x)$$

Erwartet:

$\bar{\mathrm{P}}\,(Q_5) > 0$

NELL Datenbank:

Found:

$\bar{\mathrm{P}}\,(Q_5) = 0$

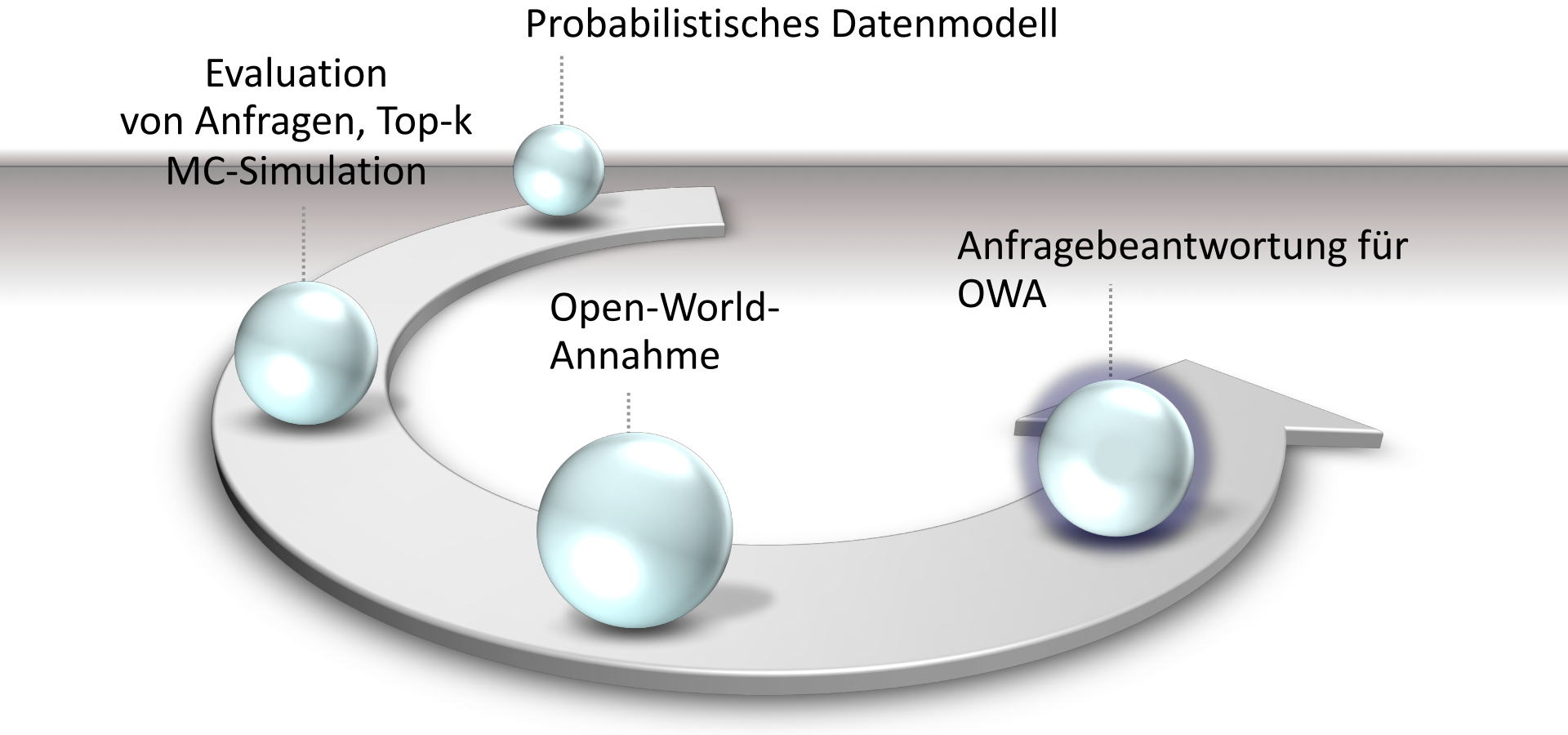
In einer offenen Welt:

Found: for $\lambda = 0.3$

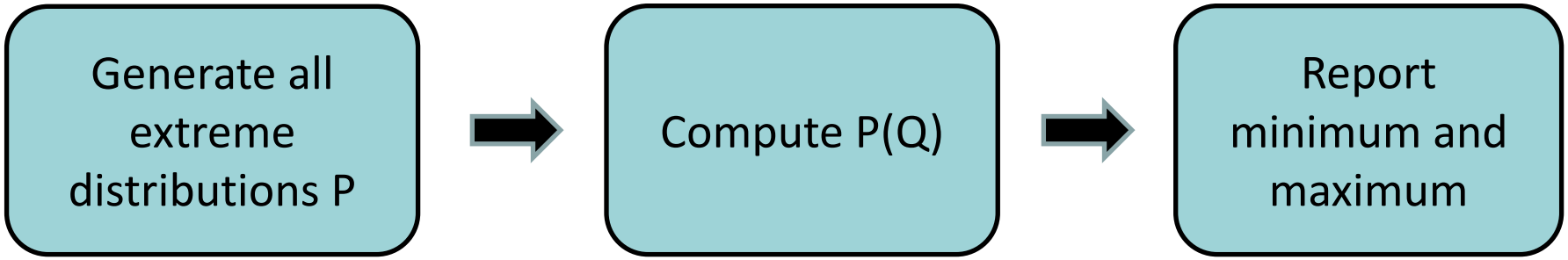
$\bar{\mathrm{P}}\,(Q_5) = 0.78$

Non-Standard-Datenbanken

Probabilistische Datenbanken



Naiver Algorithmus



Exponentiell in der Anzahl der Open-World-Tupel

Naiver Algorithmus für UCQs

Beispiel

$$\text{Exist m, a } [\underbrace{\text{Inmovie(a, m)}} \text{ AND } \underbrace{\text{Ac(a)}}]$$

- Obere Grenze: Jedes Tupel in Inmovie und Ac hat eine maximale Wahrscheinlichkeit
- Untere Grenze: Jedes Tupel in Inmovie und Ac hat eine minimale Wahrscheinlichkeit
- Monotonie für UCQs (Intervalle werden nur weiter eingeschänkt) vereinfacht Auswertung

Naiver Algorithmus for UCQs

$\underline{P}_{P\lambda}(Q)$:

Compute using a standard algorithm used for PDB

$\overline{P}_{P\lambda}(Q)$:

Construct a new PDB by adding all the open tuples with default upper probabilities λ

→

Compute using a standard algorithm used for PDB

$P(Q)$

Bei der Auswertung von $\overline{P}_{P\lambda}(Q)$ wächst die PDB exponentiell in der Domänengröße (n-stellige Relationen)

Neuer Ansatz: $Lift_O^R$ Algorithmus

CNF:
$(R(x) \vee S(y,z)) \wedge (S(x,y) \vee (\mathrm{T(x)})$

$Lift_O^R$ (Q, P, λ, D) - abbreviated by L(Q, P)

<u>Input</u>: CNF Q , probability tuples Ꝑ, threshold λ and domain D

<u>Output</u>: The upper probability $\overline{\mathrm{P}}_{(p,\lambda)}(Q)$ over domain D

<u>Step 0</u>: Base of Recursion

if Q is a single ground atom t then

if ⟨t : p⟩ ∈ Ꝑ then return p else return λ

<u>Step 1</u>: Rewriting of Query

Convert Q to union of CNFs: $Q_{UCNF} = Q_1 \vee \ldots \vee Q_m$

Example: $(R(x) \vee S(y,z)) \wedge (S(x,y) \vee (\mathrm{T(x)})$

➔ $((R(x)) \wedge (S(x,y) \vee (\mathrm{T(x)})) \vee ((S(y,z)) \wedge (S(x,y) \vee (\mathrm{T(x)}))$

$Lift_O^R$ Algorithmus

$Q_1 \perp Q_2$ if Q_1, Q_2 doesn't share any relational symbols

<u>Step 2</u>: Decomposable Disjunction

if $m > 1$ and $Q_{UCNF} = Q_1 \vee Q_2$ where $Q_1 \perp Q_2$ then

- $q_1 \leftarrow L(Q_1, P|_{Q1})$ and $q_2 \leftarrow L(Q_2, P|_{Q2})$
- return $1 - (1 - q1) \cdot (1 - q2)$

<u>Step 3</u>: Inclusion-Exclusion

if $m > 1$ but Q_{UCNF} has no independent Q_i then

- return $\sum_{s \subseteq \{1, \dots, m\}} (-1)^{|s|+1} \cdot L(\wedge_{i \in s} Q_i, P|_{\wedge_{i \in s} Q_i})$

<u>Step 4</u>: Decomposable Conjunction

if $Q = Q_1 \wedge Q_2$ where $Q_1 \perp Q_2$ then

- return $L(Q_1, P|_{Q1}) \cdot L(Q2, P|_{Q2})$

Specifically, $\mathcal{P}|_Q$ denotes the subset of $\mathcal{P}$ that talks about the predicates that appear in Q.

$Lift_O^R$ Algorithmus

Step 5: Decomposable Universal Quantifier

if Q has a separator variable x then

- let T be all constants as x-argument in P
- $q_c \leftarrow \prod_{t \in T} L(Q[x/t], Pl_{x=t})$
- $q_o \leftarrow L(Q[x/t], \emptyset)$ for some $t \in D \setminus T$
- return $q_c \cdot q_o^{|D \setminus T|}$

Step 6: Fail

A separator is a variable that appears in every atom in Q

An x-argument is an argument that hold a separator variable x

Forall a,m [Inmovie(a,m) OR Ac(a)]

Inmovie	
Pitt	Troy
Butler	300
deCaprio	Inception

Ac
Pitt
Cruise
Hayek

T = {Pitt, Butler, deCaprio, Cruise, Heyek}

$Lift_O^R$ Algorithmus für UCQs

$Lift_O^R$ berechnet die Wahrscheinlichkeiten für UCQs

<u>Daten-Komplexität:</u>

Monotone UCQs werden auf OpenPDBs in PTime evaluiert, wenn sie auf PDBs in PTime evaluiert werden und umgekehrt.

Zusammenfassung

- Einfache PDBs erfüllen nicht (mehr) die heutigen Erwartungen (Vervollständigungen nicht möglich)
- OpenPDBs setzen die Annahme der offenen Welt um
- Vorgestellt wurde ein effizienter Algorithmus zur Auswertung von UCQs in OpenPDBs
  - Ob die Semantik mit λ Bestand hat, wird die Zeit zeigen
- Nächste Schritte:
  - Erhöhung der Ausdrucksstärke (> UCQs)
  - Unendliche Domäne

Ismail Ilkan Ceylan, Adnan Darwiche and Guy Van den Broeck. Open-World Probabilistic Databases: An Abridged Report, In Proceedings of the 26th International Joint Conference on Artificial Intelligence (IJCAI), Sister Conference Best Paper Track, **2017**.

Große Frage?

- Können wir alle Verteilungen zur Beschreibung von Wahrscheinlichkeiten in dem BUD-Format darstellen?
- Sind probabilistische Datenbanken als Beschreibungssprache ausdrucksstark genug?
- Nein!