Database Schema Matching

Einführung - Ansätze - Algorithmus

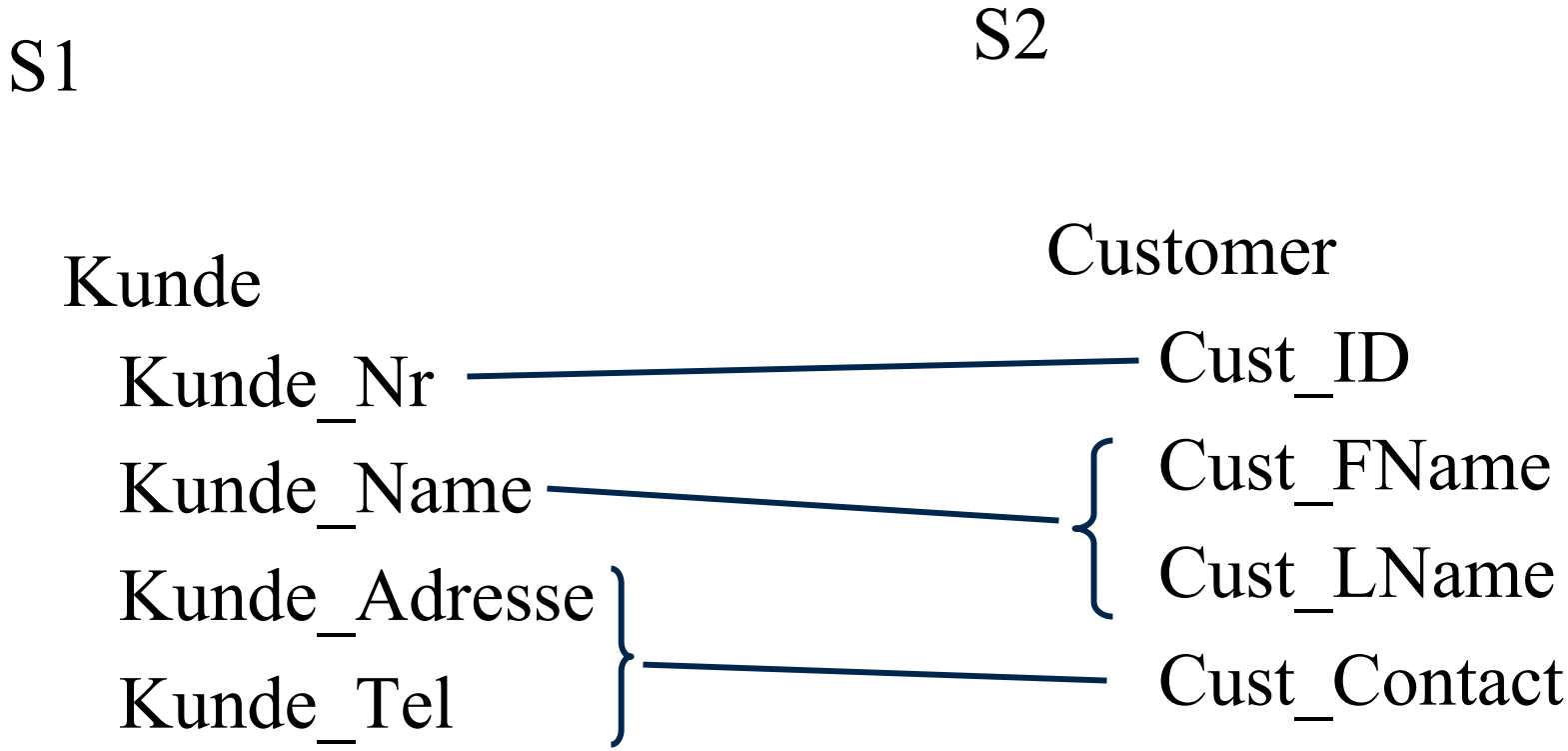
Seminar : *Semantisches Web und Agenten – WS 2006/2007*

Ovidiu Vela

Schema Matching - Definition

Identifikation von inhaltlichen Zusammenhängen zwischen verschiedenen Schemas

Schema Matching - Beispiel



Schema Matching - Motivation

- Große Schemas
  - viele Tabellen & Attribute
  - Bildschirm nicht lang genug
- Unübersichtliche Schemas
  - Tiefe Schachtelungen
  - Fremdschlüssel
  - Bildschirm nicht breit genug
- Fremde Schemas
  - Unbekannte Synonyme
  - Unbekannte Homonyme
- Fremdsprachliche Schemas
- Kryptische Schemas
  - Abkürzungen

Schema Matching - Nutzung

- **Schema Integration**
- **Data Warehouses**
- **E–Commerce**
- **Semantic Query Processing**

Schema Integration

Ziel:

Menge von Schemas in ein einziges Schema überführen

Problem:

Unabhängig entwickelte Schemas

-unterschiedliche Strukturen

-unterschiedliche Terminologie

Lösung:

Identifizierung und Charakterisierung der Strukturbeziehungen

Data Warehouses

Ziel:

Daten aus verschiedenen Datenbanken in eine umfangreiche generelle Datenbank überführen

Problem:

Umwandlung der Daten aus Quelldatenbank in dem Format für Lagerdatenbank

Lösung;

Finden der gemeinsamen Elementen der Quelldatenbank und Lagerdatenbank

E-Commerce

Ziel:

Das Ermöglichen des Informationsaustausches zwischen Systemen

Problem:

Übersetzung verschiedenen Nachrichten

- unterschiedliche Namen
- unterschiedliche Datentypen
- unterschiedliche Strukturen

Lösung:

Matching zwischen Nachrichtenschemas

Semantic Query Processing

Ziel:

Benutzer soll mit dem System in natürlicher Sprache kommunizieren

Problem:

Die Benutzerkonzepte sind verschieden von denen vom System benutzte Konzepte

Lösung:

Abbildung zwischen den Elementen des Systems und die Konzepten die in der Frage vorkommen

Schema Matching -Algorithmus

- EINGABE
  - zwei Schemata mit Attributmengen A und B
- IDEE
  - Bilde Kreuzprodukt aller Attribute aus A und B
  - Für jedes Paar berechne Ähnlichkeit
    - Z.B. bzgl. Attributnamen
    - Z.B. bzgl. gespeicherten Daten
- AUSGABE
  - Paare mit Ähnlichkeit > Schwellwert (Mapping)

Schema Matching - Ausgabe

- Abbildung – Menge von Abbildungselementen
  - Abbildungselement - Paar von Elementen aus zwei Schemas
    - (a, b) von Typ 1:1, 1:n, n:1, n:m
    - Abbildungsausdruck
  - Abbildungsausdruck – Art der Relation zwischen Elementen
    - Gerichtet oder ungerichtet
    - Einfache Relation (=, >, ...)
    - Funktionen (Konkatenation, Addition, ...)
    - ER-Relationen (is-a, part-of, ...)
    - Mengenorientiert (Durschnitt, ...)

Ausgabe - Beispiel

({S1.KundeNr}, {S2.CustId})

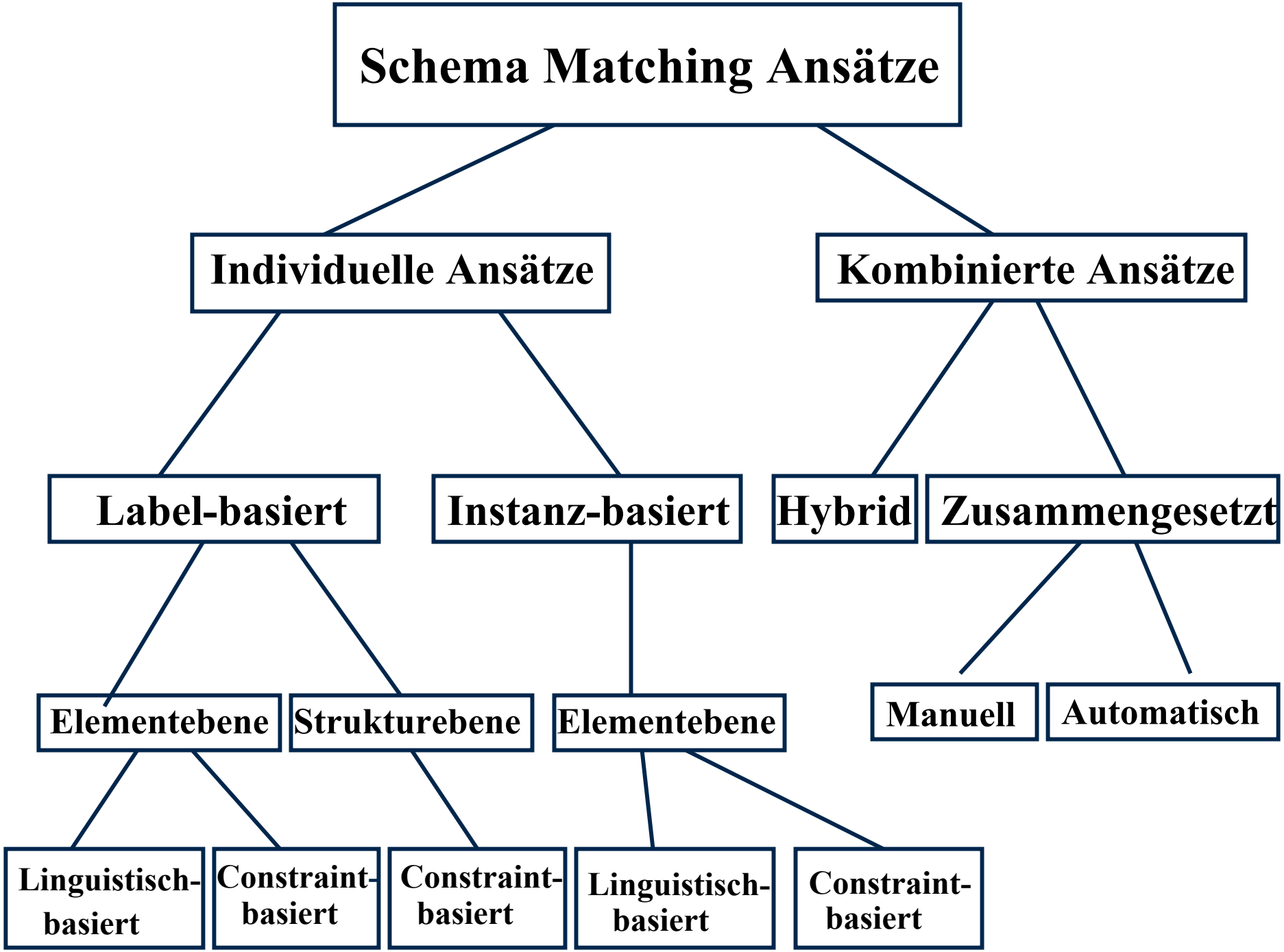
S1.KundeNr = S2.CustId

(S1.KundeName, {S2.CustFName, S2.CustLName})

S1.KundeName = Concatenate(S2.CustFName, S2.CustLName)

({S1.KundeAdr, S1.KundeTel}, S2.CustContact)

Concatenate(S1.KundeAdr, S1.KundeTel) = S2.CustContact


Schema Matching Ansätze
Individuelle Ansätze
Kombinierte Ansätze
Label-basiert
Instanz-basiert
Hybrid
Zusammengesetzt
Elementebene
Strukturebene
Elementebene
Manuell
Automatisch
Linguistisch-basiert
Constraint-basiert
Constraint-basiert
Linguistisch-basiert
Constraint-basiert

Schema Matching - Tools

- **SemInt** (Northwestern Univ.)
  - ER, Label & Instanz basiert, [1,1], Hybrid
- **LSD** (Univ. of Washington)
  - XML, Label & Instanz basiert, [1,1], Zusammengesetzt
- **SKAT** (Standford Univ.)
  - XML, IDL, [1,1][n,1], Label basiert, Hybrid
- **TransScm** (Tel Aviv Univ.)
  - SGML, OO, [1,1], Label basiert, Hybrid
- **DIKE** (Univ. Of Reggio Calabria, Univ of Calabria)
  - ER, [1,1], Label basiert, Hybrid
- **ARTEMIS** (Univ. Of Milano, Univ. of Brescia)
  - ER, OO, [1,1], Label basiert, Hybrid
- **Cupid** (Microsoft Research)
  - XML, ER , [1,1][n,1], Label basiert, Hybrid

Similarity Flooding

- Entwickelt von Sergey Melnik, Hector Garcia-Molina (Stanford), Erhard Rahm (Leipzig)
- Matching Algorithmus, um Ähnlichkeit zwischen Knoten zweier gerichteten und beschrifteten Graphen zu bestimmen
- Iterative Berechnung eines Fixpoint
- Andwendbar auf diverse Datenstrukturen (Modelle)

Similarity Flooding

- Die Idee
  - Umwandlung der Modelle in gerichtete Graphen
  - Benutze diese Graphen für eine iterative Fixpoint Berechnung der Ähnlichkeit
    - Zwei Knoten (aus zwei Graphen) sind ähnlich wenn ihre benachbarte Knoten ähnlich sind

      (Die Ähnlichkeit zweier Knoten flutet die Ähnlichkeit der benachbarten Knoten)

Matching - Algorithmus

```
CREATE TABLE Personnel (
        Pno int,
        Pname string,
        Dept string,
        Born date,
        UNIQUE pkey(Pno) );
```

S1

```
CREATE TABLE Employee (
        EmpNo int PRIMARY KEY,
        EmpName varchar(50),
        DeptNo int REFERENCES
                        Department,
        Salary dec(15,2),
        Birthdate date ) ;
CREATE TABLE Department (
        DeptNo int PRIMARY KEY,
        DeptName varchar(70) );
```

S2

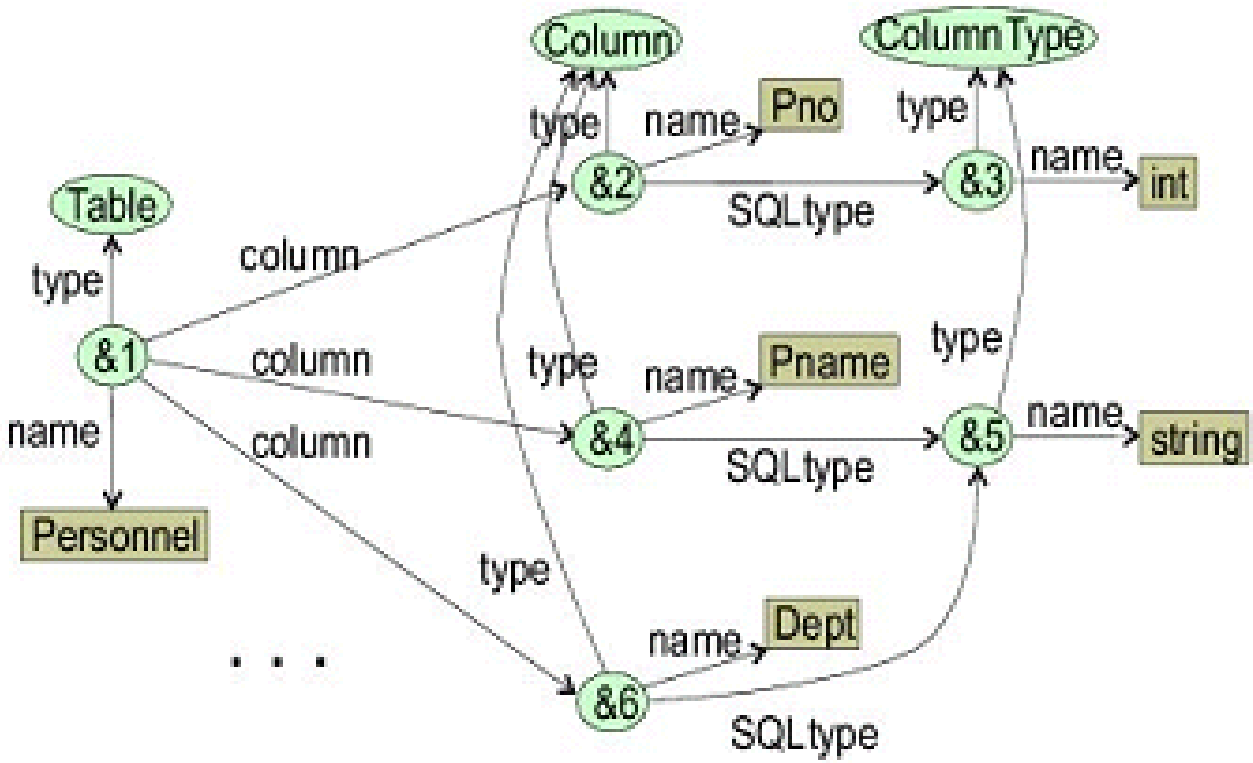
- 1. G1 = SQL2Graph(S1);
  G2 = SQL2Graph(S2);
- 2. initialMap = StringMatch(G1, G2);
- 3. product = SFJoin(G1, G2, initialMap);
- 4. result = SelectThreshold(product);

Matching - Algorithmus

Schritt 1 – Umwandlung der Schemas in Graphen

G1 = SQL2Graph(S1);

CREATE TABLE
Personnel (
Pno int,
Pname string,
Dept string,
Born date,
UNIQUE *pkey*(*Pno*));



Matching - Algorithmus

Similarity	Node in G1	Node in G2
1.0	Column	Column
0.66	Column	TypeColumn
0.66	Dept	DeptNo
0.66	Dept	DeptName
0.50	UniqueKey	PrimaryKey
0.26	Pname	DeptName
0.26	Pname	EmpName
0.22	date	BirthDate
0.11	Dept	Department
0.06	int	Department

Schritt 2 – Grobes Matching

Namen-basiert

initialMap= StringMatch(G1, G2);

Matching - Algorithmus

- Schritt 3 – Similarity Flooding(SFJoin)

  product = SFJoin(G1, G2, initialMap);

  - Die Anfangswerte von initialMap
  - Iteration – in jedem Schritt hat die Ähnlichkeit zweier Elemente eine Auswirkung auf die Ähnlichkeit ihrer Nachbarn
  - Solange iterieren bis die Ähnlichkeitswerte sich stabilisieren – der Fixpoint ist erreicht

Matching - Algorithmus

- Schritt 4 – Filterung

  result = SelectThreshold(product);

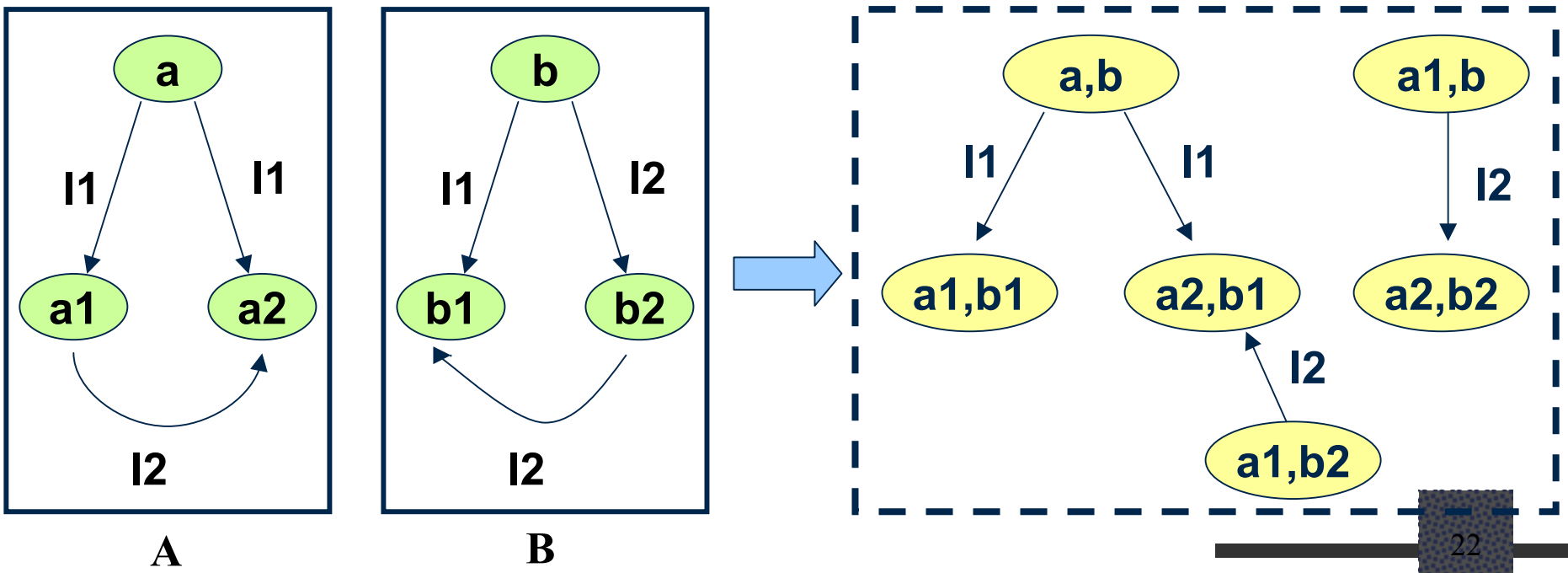
Similarity	Node in G1	Node in G2
1.0	Column	Column
0.81	[Table:Personnel]	[Table:Employee]
0.66	ColumnType	ColumnType
0.44	[ColumnType:int]	[ColumnType:int]
0.43	Table	Table
0.35	[ColumnType:date]	[ColumnType:date]
0.29	[UniqueKey:pkey]	[PrimaryKey: on EmpNo]
0.28	[Col:Personnel/Dept]	[Col:Departament/DeptName]
0.25	[Col:Personnel/Pno]	[Col:Employee/EmpNo]
0.19	UniqueKey	PrimaryKey
0.18	[Col:Personnel/Pname]	[Col:Employee/EmpName]
0.17	[Col:Personnel7Born]	[Col:Employee/Birthdate]

Similarity Flooding - Algorithmendetails

product = SFJoin(G1, G2, initialMap);

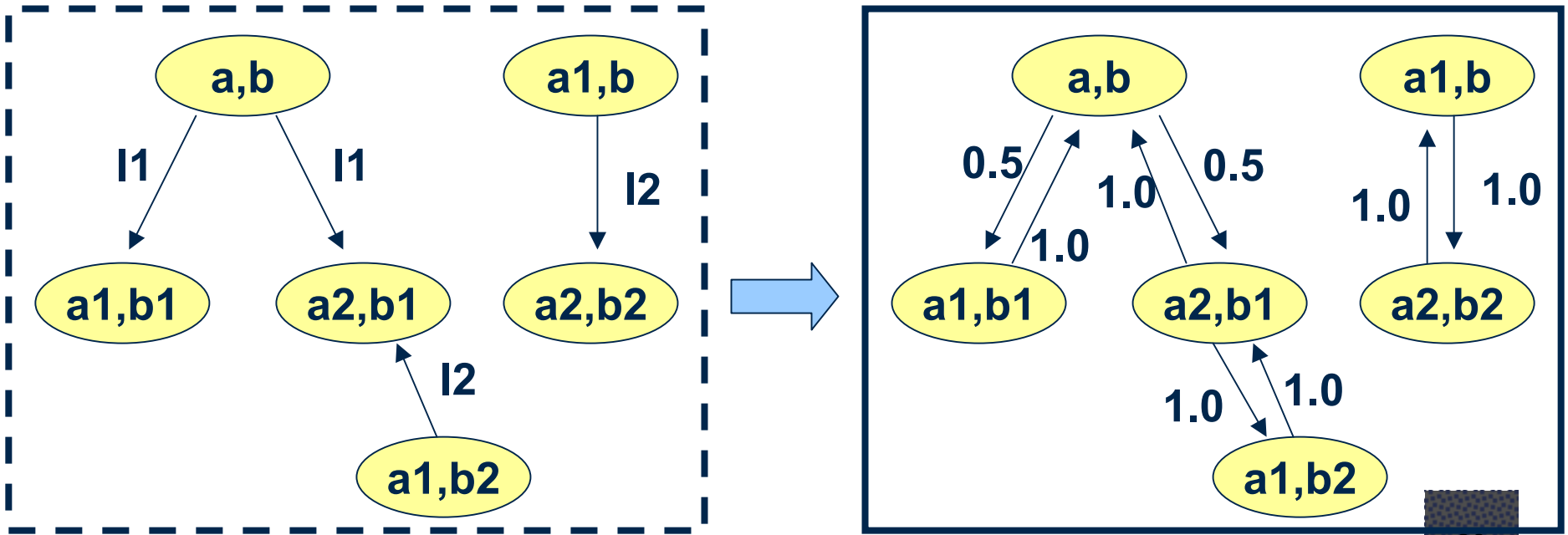
- *Paarweise Konnektivitätsgraph(PCG)*

$$((x, y), e, (x', y')) \in \boldsymbol{PCG}(A, B) \Leftrightarrow (x, e, x') \in A \textit{ and } (y, e, y') \in B$$



Similarity Flooding - Algorithmendetails

- Induzierte Propagationsgraph
  - Einführung der Kanten in Gegenrichtung
  - Propagierungskoeffizient für jede Kante



Similarity Flooding - Algorithmendetails

- Fixpoint Berechnung

Ähnlichkeitsfunktion $\sigma(x,y) \geq 0$ *für alle* $x \in A,\ b \in B$

Iterative Berechnung der σ Funktion

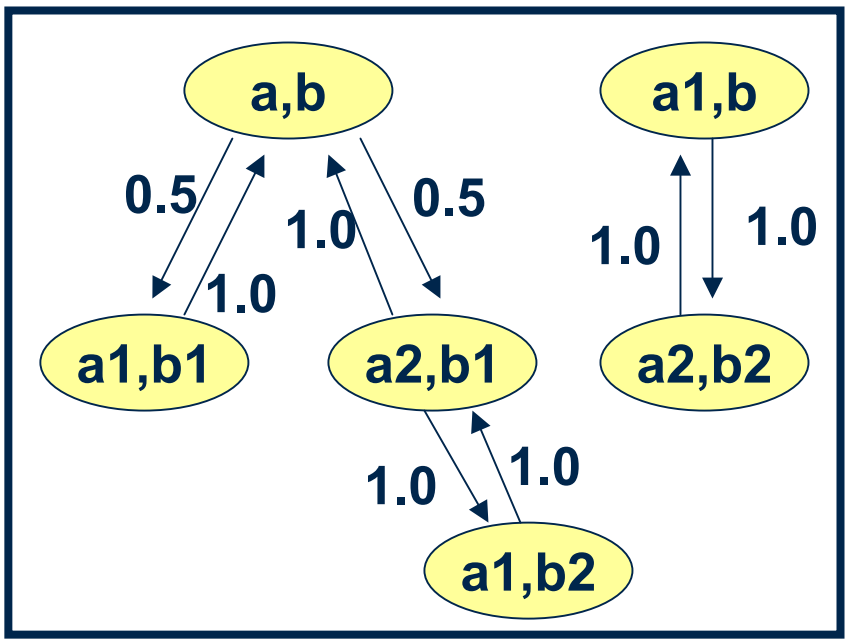
$$\sigma^{i+1}(x,y) = \sigma^{i}(x,y) + \sum_{(a_u,p,x)\in A,(b_u,p,y)\in B} \sigma^{i}(a_u,b_u)\cdot\omega((a_u,b_u),(x,y)) + \sum_{(x,p,a_v)\in A,(y,p,b_v)\in B} \sigma^{i}(a_v,b_v)\cdot\omega((a_v,b_v),(x,y))$$

σ^{0} = Ähnlichkeitwerten von Initial Maping

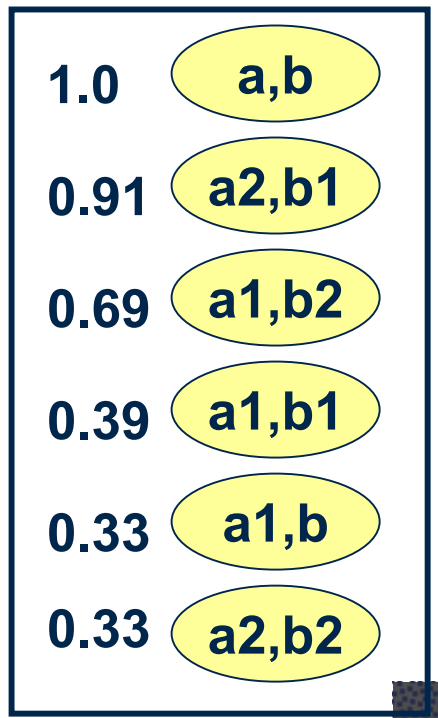
Normalisierung der Werte durch Teilung mit der maximale σ

Similarity Flooding - Algorithmendetails

- Fixpoint Berechnung



nach 5 Iterationen



Similarity Flooding - Algorithmendetails

- Fixpoint Berechnung

  Iteration bis euklidische Distanz des residual Vektors $\Delta(\sigma^{n}, \sigma^{n-1})$ kleiner als ein ε

  Mehrere Formeln für Fixpoint Berechnung

$$\text{Basic} : \sigma^{i+1} = \mathit{normalize}\ (\sigma^{i} + \varphi(\sigma^{i}))$$

$$\text{A} : \sigma^{i+1} = \mathit{normalize}\ (\sigma^{0} + \varphi(\sigma^{i}))$$

$$\text{B} : \sigma^{i+1} = \mathit{normalize}\ (\varphi(\sigma^{0} + \sigma^{i}))$$

$$\text{C} : \sigma^{i+1} = \mathit{normalize}\ (\sigma^{0} + \sigma^{i} + \varphi(\sigma^{0} + \sigma^{i}))$$

Similarity Flooding - Filterung

- Für jedes Element eine Menge von Matchkandidaten
- Von n Matchingpaare 2^n Abbildungsuntermengen
- Auswahlstrategie einer Untermenge
  1. Model-spezifische Constraints
     - Typ Constraints
     - Kardinalität
  2. Auswahlverfahren entwickelt für bipartite Graphen
     - Stable marriage
     - Maximal matcing
  3. Anwendung der Auswahlverfahren auf Modellklassen und Auswertung der Ergebnisse der Auswahlverfahren

Similarity Flooding - Filterung

- Mapping = ungerichteter gewichteter bipartiter Graph
- Ansätze aus dem Gebiet der bipartiten Graphen
  - Stable marriage

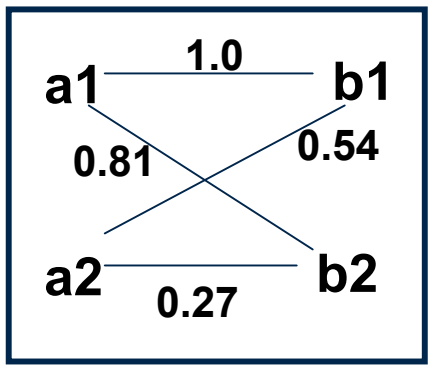
    $\nexists$(a,b) und (a‘,b‘) so dass σ (a,b‘) > σ (a,b) und σ (b‘,a) > σ(b‘,a‘)
  - Assignment Problem

    $\sum$ σ(a,b) maximal
  - Perfectionist egalitarian polygamy

    $\nexists$(a,b) und a‘ b‘ so dass σ (a,b) < σ (a,b‘) oder σ (b,a) < σ (b,a‘)
  - Maximum matching, complet matching, etc.

Similarity Flooding - Filterung

Mapping M



4 Paare => 16 Mögliche Untermengen

[1,1]-[1,1]

Kardinalität constraint

M1 = {(a1,b1),(a2,b2)}

M2 = {(a1,b2),(a2,b1)}

Auswahlverfahren

$\sum_{M1} \sigma = 1.27$, $\sum_{M2} \sigma = 1.35$

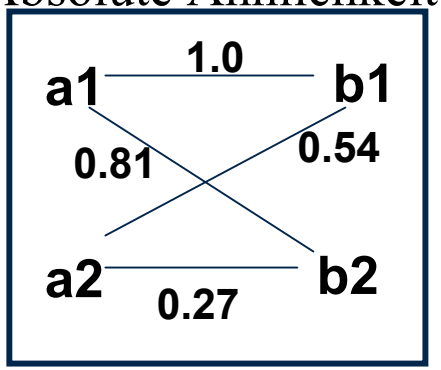
M1 stable marriage

Similarity Flooding - Filterung

SelectThreshold Operator

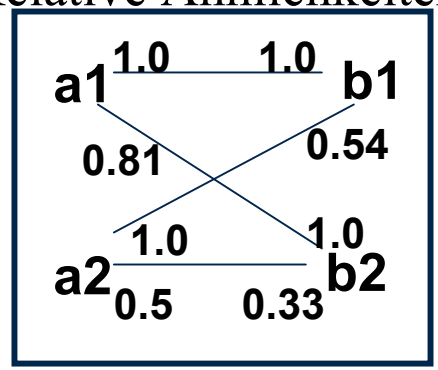
Berechnung der relativen Ähnlichkeiten

Absolute Ähnlichkeiten



σ‘

Relative Ähnlichkeiten



$$\sigma'(a) = \max_{b \in B}(\sigma(a,b))$$

$$\sigma'(a,b) = \frac{\sigma(a,b)}{\sigma'(a)}$$

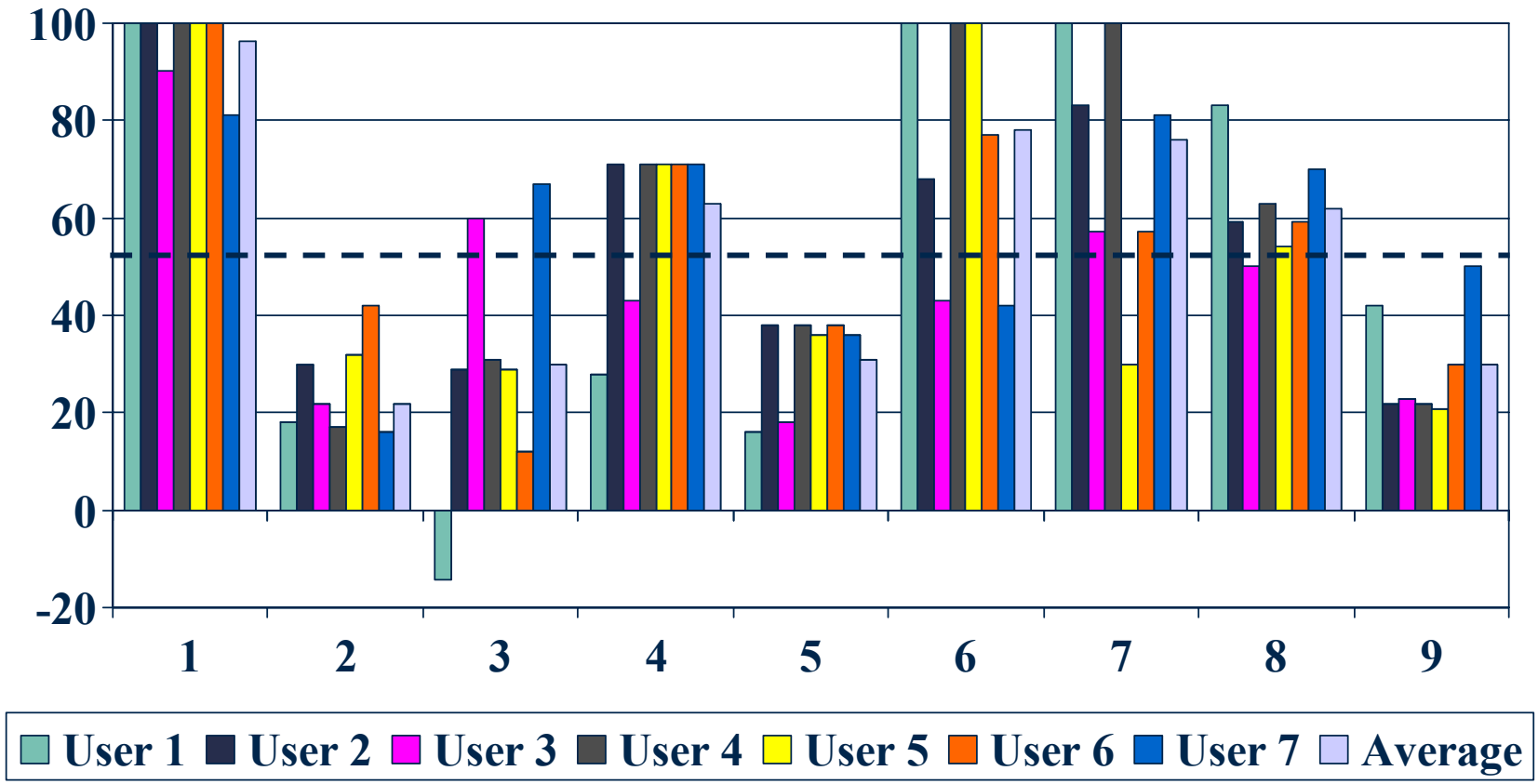
Auswahl der Paare mit relativer Ähnlichkeit > Schwelle t
Auswahl einer Menge unter stable mariage Voraussetzungen

Similarity Flooding - Evaluierung

- 9 relativ einfache Abbildungsprobleme
  - Abbildung von XML Schemas (1,2,3)
  - Abbildung von XML Schemas mit Instanzen (4,5,6)
  - Abbildung von relationale Schemas (7,8,9)

  (Propagationsgraphen zwischen 128 und 1222 Knoten)
- 7 Benutzer
- Fixpoint Formel C - $\text{normalize}(\sigma^0 + \sigma^i + \varphi(\sigma^0 + \sigma^i))$
- SelectThreshold Operator mit t = 1

Similarity Flooding - Evaluierung



Similarity Flooding - Evaluierung

- Filterung
  - ***Thresold*** - SelectThreshold Operator t=1, [0,n]-[0,n]
  - ***Exact*** - Threshold mit [0,1]-[0,1]
  - ***Best*** – Assignment Problem [0,1]-[0,1]
  - ***Left*** – Assignment Problem [0,1]-[1,1]
  - ***Right*** – Assignment Problem [1,1]-[0,1]
  - ***Outer*** – [1,n]-[1,n]
- Fixpoint Formel

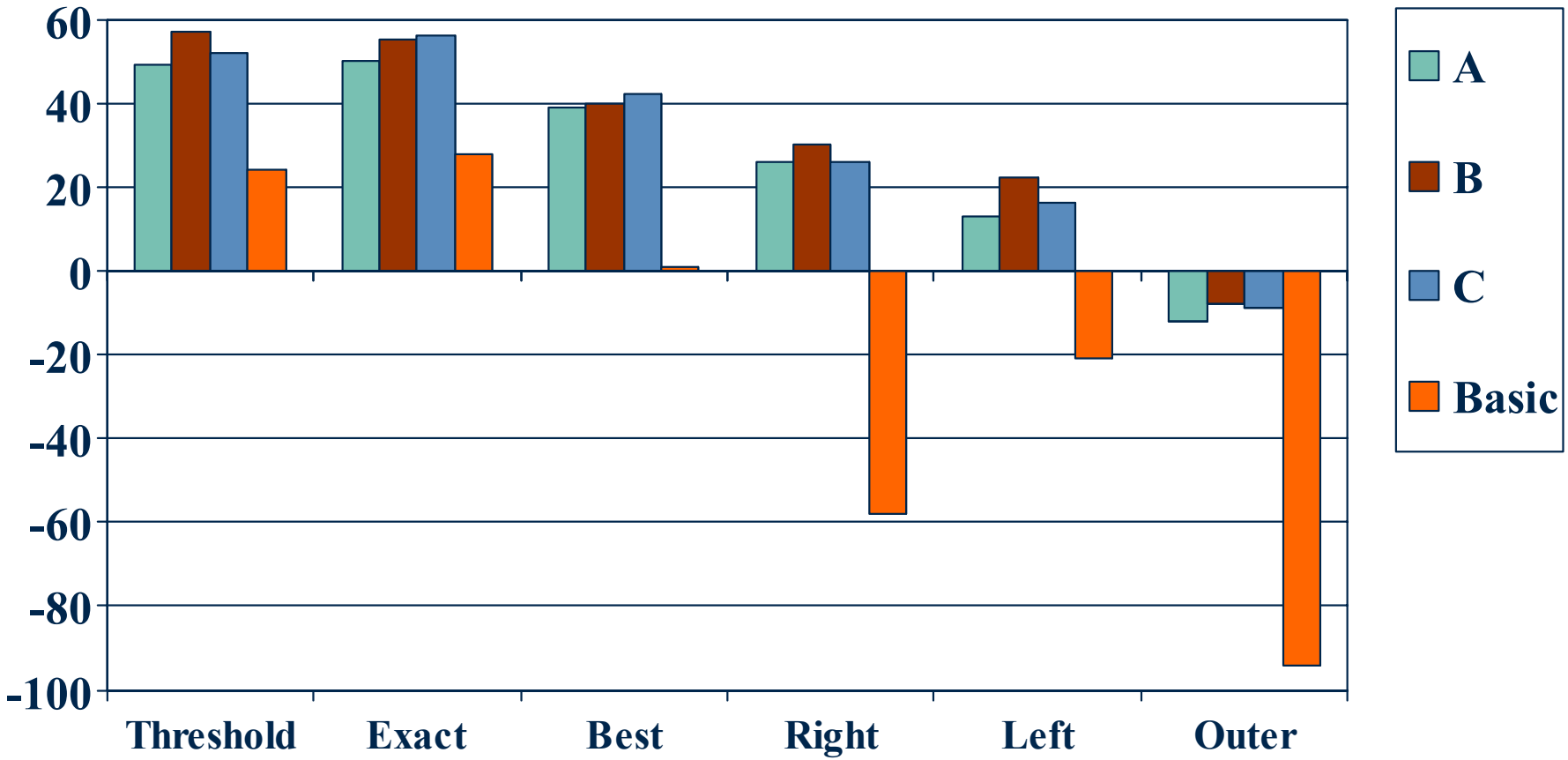
$$\textbf{Basic}: \sigma^{i+1} = normalize(\sigma^i + \varphi(\sigma^i))$$

$$\textbf{A}: \sigma^{i+1} = normalize(\sigma^0 + \varphi(\sigma^i))$$

$$\textbf{B}: \sigma^{i+1} = normalize(\varphi(\sigma^0 + \sigma^i))$$

$$\textbf{C}: \sigma^{i+1} = normalize(\sigma^0 + \sigma^i + \varphi(\sigma^0 + \sigma^i))$$

Similarity Flooding - Evaluierung



Similarity Flooding – Pro & Kontra

- Pro
  - Innovative Methode (2001-2002)
  - Für jede Schematyp anwendbar
  - Keine Trainingsphase notwendig
  - Flexibel was Filterung betrifft
- Kontra
  - Schlechte Grundlage für Propagierung
  - Fehler sind auch propagiert
  - Flooding Algorithmen sind üblicherweise langsam
  - Benutzt keine Instanzen
  - Kann nicht komplexe Relationen zwischen Elemente erkennen
  - InitialMatch hat sehr grossen Einfluss auf das Ergebniss, was wieder zu der Frage führt: Wie kann man ein gutes Matching entwerfen?