Approximate Joins: Concepts and Techniques

Nick Koudas (University of Toronto)

and

Divesh Srivastava (AT&T Labs-Research)

Outline

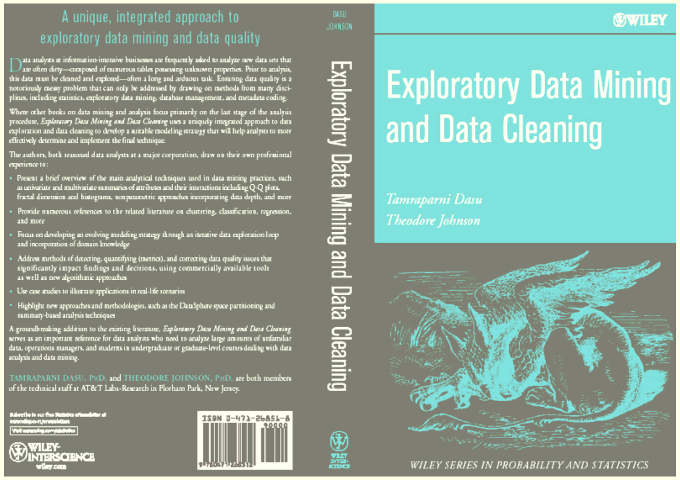
- Part I: Motivation (15 min)
  - Data quality, applications
  - Approximate joins
  - Fellegi-Sunter
- Part II: Similarity metrics (35 min)
- Part III: Efficient algorithms (40 min)

Data Quality: Status

- Pervasive problem in large databases
  - Inconsistency with reality: 2% of records obsolete in customer files in 1 month (deaths, name changes, etc) [DWI02]
  - Pricing anomalies : UA tickets selling for $5, 1GB of memory selling for $19.99 at amazon.com
- Massive financial impact
  - $611B/year loss in US due to poor customer data [DWI02]
  - $2.5B/year loss due to incorrect prices in retail DBs [E00]
- Commercial tools: specialized, rule-based, programmatic

How are Such Problems Created?

- Human factors
  - Incorrect data entry
  - Ambiguity during data transformations
- Application factors
  - Erroneous applications populating databases
  - Faulty database design (constraints not enforced)
- Obsolence
  - Real-world is dynamic



Application: Merging Lists

- Application: merge address lists (customer lists, company lists) to avoid redundancy
- Current status: “standardize”, different values treated as distinct for analysis
  - Lot of heterogeneity
  - Need approximate joins
- Relevant technologies
  - Approximate joins
  - Clustering

ADDR
11810 WILLS RD ALPHARETTA GA 30076
11810 WILLS ROAD ALPHARETTA GA30004
FLR 110 RM 155 11810 WILLS RD ALPHARETTA GA 300042055
FLR 110 RM MAIN 11810 WILLIS RD ALPHARETTA GA 300042055
FLR 110 RM MAIN 11810 WILLS RD ALPHARETTA GA 30076
FLR BLDG 110 RM 155 11810 WILLS RD ALPHARETTA GA 300042055
FLR BLDG 110 RM MAIN 111810 WILLS RD ALPHARETTA GA 30076
FLR BLDG 110 RM MAIN 11810 WILLS RD ALPHARETTA GA 300042055
FLR MAIN RM 110 11810 WILLS RD ALPHARETTA GA 300042055
FLR MAIN RM BLGD 110 11810 WILLS RD ALPHARETTA GA 300042055
FLR NA RM NA 11810 WILLS RD ALPHARETTA GA 300042055
BLDG 110 FLR 1 RM 110 11810 WILLIS RD ALPHARETTA GA 30004205
BLDG 110 FLR 1 RM 110 11810 WILLS RD ALPHARETTA GA 300042055
BLDG 110 FLR 1 RM 110 11810 WILLS RD ALPHARETTA GA 300042081
BLDG 110 FLR 1 RM 110 11810 WILLS RD ALPHARETTA GA 30076
BLDG 110 FLR 1 RM RING 11810 WILLS RD ALPHARETTA GA 30004208
FLR 1 RM 1 11810 WILLS RD ALPHARETTA GA 300042055
FLR 1 RM 110 11801 WILLIS RD ALPHARETTA GA 30076
FLR 1 RM 110 11810 WILLIS RD ALPHARETTA GA 300042055
FLR 1 RM 110 11810 WILLIS RD ALPHARETTA GA 30076
FLR 1 RM 110 11810 WILLS RD ALPHARETTA GA 300042055
FLR 1 RM 110 11810 WILLS RD ALPHARETTA GA 300042081
FLR 1 RM 110 11810 WILLS RD ALPHARETTA GA 30076
FLR 1 RM 110 11810 WILLS RD ATLANTA GA 30076
FLR 1 RM 110 11810 WILLS ROAD ALPHARETTA GA 30076
FLR 1 RM BLDG 110 11810 WILLS RD ALPHARETTA GA 300042055
FLR 1 RM BLDG 11810 WILLS RD ALPHARETTA GA 300042055
FLR 1 RM COMPUTER 11810 WILLS RD ALPHARETTA GA 300042055

Application: Homeland Security

- Application: correlate airline passenger data with homeland security data for no-fly lists
- Current status: "match" on name, deny boarding
  - Use more match attributes
  - Obtain more information
- Relevant technologies
  - Schema mappings
  - Approximate joins

TRAVEL

'No-fly list' keeps infants off planes

Tuesday, August 16, 2005; Posted: 4:49 a.m. EDT (08:49 GMT)

WASHINGTON (AP) -- Infants have been stopped from boarding planes at airports throughout the United States because their names are the same as or similar to those of possible terrorists on the government's "no-fly list."

It sounds like a joke, but it's not funny to parents who miss flights while scrambling to have babies' passports and other documents faxed.

Ingrid Sanden's 1-year-old daughter was stopped in Phoenix, Arizona, before boarding a flight home to Washington at Thanksgiving.

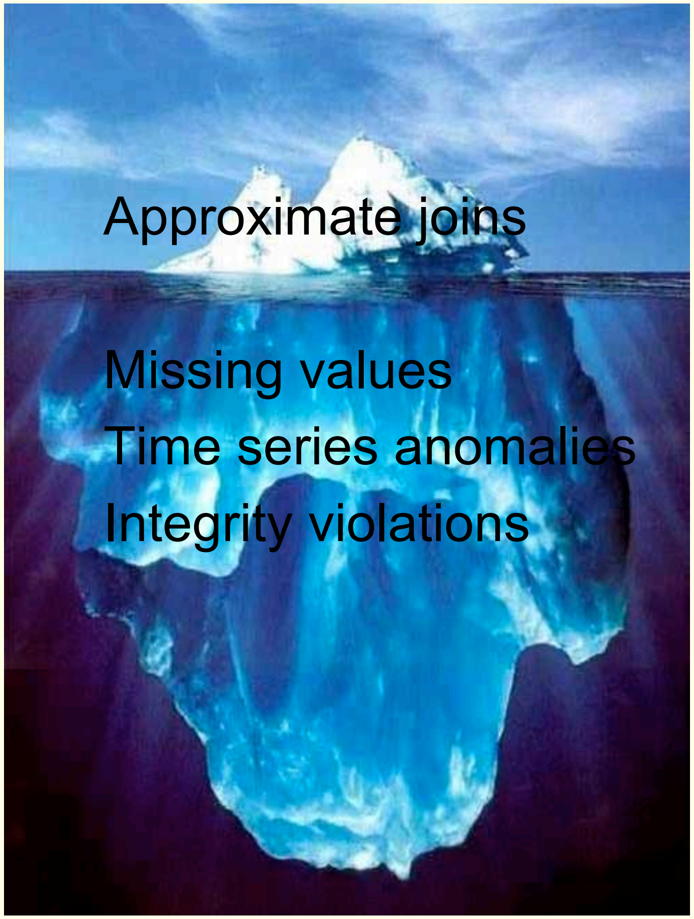
"I completely understand the war on

(AP PHOTO)

Ingrid Sanden holds her 1-year-old daughter, who was stopped before boarding a flight last Thanksgiving.

Approximate Joins: Tip of the Iceberg

- An approximate join of R_1 and R_2 is
  - A subset of the cartesian product of R_1 and R_2
  - "Matching" specified attributes of R_1 and R_2
  - Labeled with a similarity score > t > 0
- Related terms: record linkage, co-reference resolution, deduplication



The Fellegi-Sunter Model [FS69]

- Formalized the approach of Newcombe et al., [NKAJ59]
- Given two sets of records (relations) A and B perform an approximate join
  - $A \times B = \{(a,b) \mid a \in A, b \in B\} = M \cup U$
  - $M = \{(a,b) \mid a=b, a \in A, b \in B\}$; matched
  - $U = \{(a,b) \mid a <> b, a \in A, b \in B\}$; unmatched
- $\gamma(a,b) = (\gamma^i(a,b))$ i=1..K comparison vector
  - Contains comparison features e.g., same last names, same SSN, etc.
- Γ: range of $\gamma(a,b)$ the comparison space.

The Fellegi-Sunter Model

- Seeking to characterize (a,b) as
  - A_1 : match ; A_2 : uncertain ; A_3 : non-match
- Function (linkage rule) from Γ to $\{A_1\ A_2\ A_3\}$
- Distribution D over A x B
  - $m(\gamma) = P(\gamma(a,b) \mid (a,b) \in M\}$
  - $u(\gamma) = P(\gamma(a,b) \mid (a,b) \in U\}$

Fellegi-Sunter Result

- Sort vectors γ by m (γ)/u (γ) non increasing order; choose n < n'
  - μ = $\sum_{i=1}^{n} u(\gamma_\iota)$ λ = $\sum_{i=n'}^{N} m(\gamma_\iota)$
  - Linkage rule with respect to minimizing $P(A_2)$, with $P(A_1|U) = \mu$ and $P(A_3|M) = \lambda$ is
    - $\gamma_1,\ldots\ldots\ldots,\gamma_n,\gamma_{n+1},\ldots\ldots\ldots,\gamma_{n'-1},\gamma_{n'},\ldots\ldots\ldots,\gamma_N$
    - A_1 A_2 A_3
    - Intuition
      - Swap i-th vector declared as A_1 with j-th vector in A_2
      - If $u(\gamma_i) = u(\gamma_j)$ then $m(\gamma_j) < m(\gamma_I)$
      - After the swap, $P(A_2)$ is increased

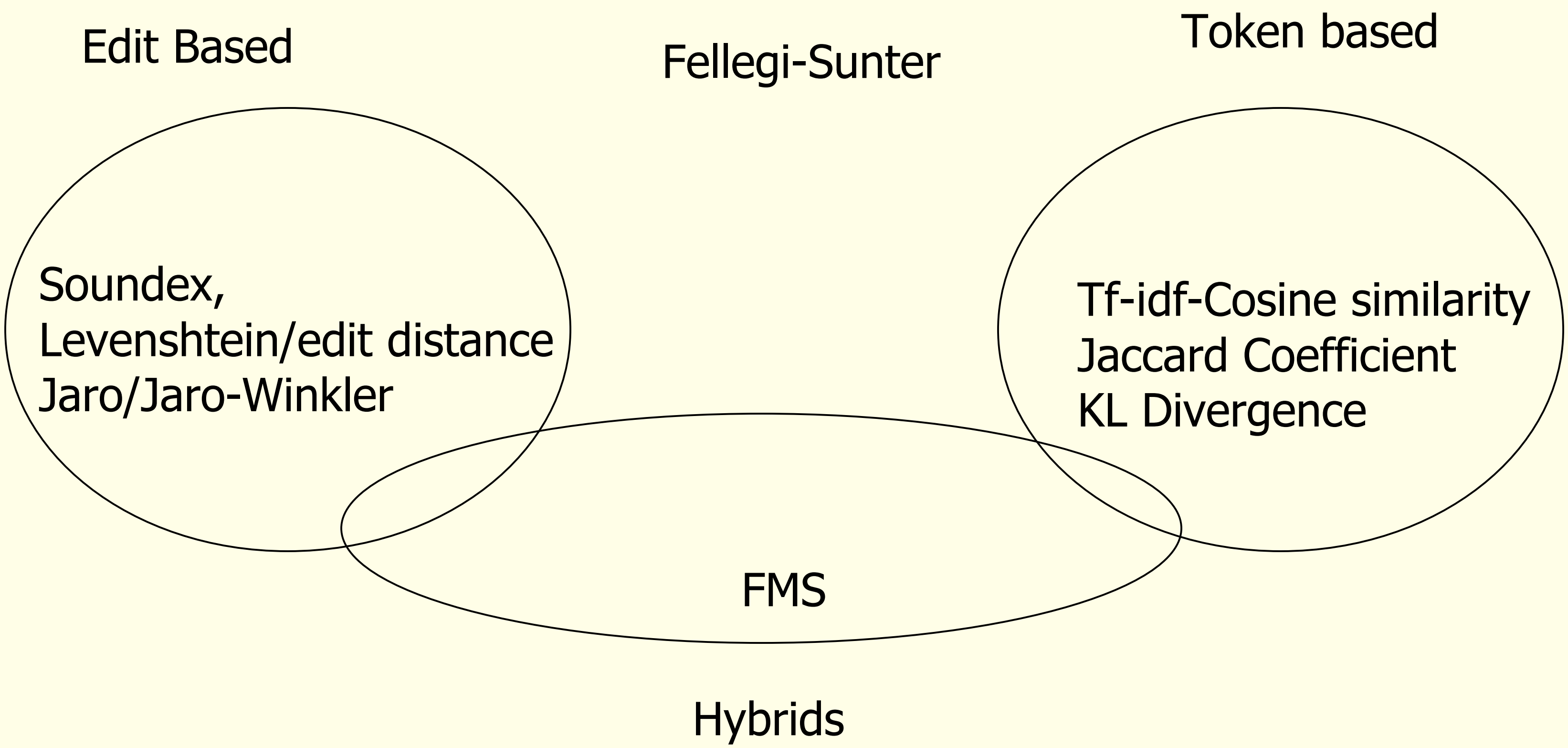
Fellegi-Sunter Issues:

- Tuning:
  - Estimates for m (γ), u (γ) ?
    - Training data: active learning for M, U labels
    - Semi or un-supervised clustering: identify M U clusters
  - Setting μ , λ?
  - Defining the comparison space Γ?
    - Distance metrics between records/fields
- Efficiency/Scalability
  - Is there a way to avoid quadratic behavior (computing all |A|x|B| pairs)?

Outline

- Part I: Motivation (15 min)
- Part II: Similarity metrics (35 min)
  - Edit metrics
  - Token based metrics
  - Hybrid metrics
- Part III: Efficient algorithms (40 min)

Classification of the measures



Attribute Standardization

- Several attribute fields in relations have loose or anticipated structure:
  - Addresses, names
  - Bibliographic entries (mainly for web data)
- Preprocessing to standardize such fields
  - Enforce common abbreviations, titles
  - Extract structure from addresses
- Part of ETL tools, commonly using field segmentation and dictionaries
- Recently machine learning approaches
  - HMM encode universe of states [CCZ02]

Field Similarity

- Application notion of 'field'
  - Relational attribute, set of attributes, entire tuples.
- Basic problem: given two field values quantify their 'similarity' (wlog) in [0..1].
- If numeric fields, use numeric methods.
- Problem challenging for strings.

Soundex Encoding

- A phonetic algorithm that indexes names by their sounds when pronounced in english.
- Consists of the first letter of the name followed by three numbers. Numbers encode similar sounding consonants.
  - Remove all W, H
  - B, F, P, V encoded as 1, C,G,J,K,Q,S,X,Z as 2
  - D,T as 3, L as 4, M,N as 5, R as 6, Remove vowels
  - Concatenate first letter of string with first 3 numerals
- Ex: great and grate become 6EA3 and 6A3E and then G63
- More recent, metaphone, double metaphone etc.

Edit Distance [G98]

- Character Operations: I (insert), D (delete), R (Replace).
- Unit costs.
- Given two strings, s,t, edit(s,t):
  - Minimum cost sequence of operations to transform s to t.
  - Example: edit(Error,Eror) = 1, edit(great,grate) = 2
- Folklore dynamic programming algorithm to compute edit();
- Computation and decision problem: quadratic (on string length) in the worst case.

Edit Distance

- Several variants (weighted, block etc) -- problem can become NP-complete easily.
- Operation costs can be learned from the source
  - String alignment = sequence of edit operations emitted by a memoryless process [RY97].
- Observations
  - May be costly operation for large strings
  - Suitable for common typing mistakes
    - Comprehensive vs Comprenhensive
  - Problematic for specific domains
    - AT&T Corporation vs AT&T Corp
    - IBM Corporation vs AT&T Corporation

Jaro Rule [J89]

- Given strings $s = a_1,\ldots,a_k$ and $t = b_1,\ldots,b_L$ a_i in s is common to a character in t if there is a b_j in t such that $a_i = b_j$ $i\text{-}H \leq j \leq i+H$ where
  - $H = \min(|s|,|t|)/2$
- Let $s' = a_1',\ldots,a_{k'}'$ and $t' = b_1',\ldots,b_{L'}'$ characters in s (t) common with t (s)
- A transposition for s',t' is a position i such that $a_i' \neq b_i'$.
- Let $T_{s',t'}$ be half the number of transpositions in s' and t'.

Jaro Rule

- Jaro(s,t) = $\frac{1}{3}\left(\frac{|s'|}{|s|}+\frac{|t'|}{|t|}+\frac{|s'|-T_{s',t'}}{|s'|}\right)$
- Example:
  - Martha vs Marhta
    - H = 3, s' = Martha, t' = Marhta, $T_{s',t'}$ = 1
    - Jaro(Martha,Marhta) = 0.9722
  - Jonathan vs Janathon
    - H = 4, s' = jnathn, t' = jnathn, $T_{s',t'}$ = 0
    - Jaro(Jonathan,Janathon) = 0.5

Jaro-Winkler Rule [W99]

- Uses the length P of the longest common prefix of s and t; P' = max(P,4)
- Jaro-Winkler(s,t) = $Jaro(s,t) + \frac{P'}{10}(1 - Jaro(s,t))$
- Example:
  - JW(Martha,Marhta) = 0.9833
  - JW(Jonathan,Janathon) = 0.7
- Observations:
  - Both intended for small length strings (first,last names)

Term (token) based

- Varying semantics of ‘term’
  - Words in a field
    - ‘AT&T Corporation’ -> ‘AT&T’ , ‘Corporation’
  - Q-grams (sequence of q-characters in a field)
    - {‘AT&’,’T&T’,’&T ‘, ‘T C’,’ Co’,’orp’,’rpo’,’por’,’ora’,’rat’,’ati’,’tio’,’ion’} 3-grams
- Assess similarity by manipulating sets of terms.

Overlap metrics

- Given two sets of terms S, T
  - Jaccard coef.: Jaccard(S,T) = $|S \cap T| / |S \cup T|$
  - Variants
    - If scores (weights) available for each term (element in the set) compute Jaccard() only for terms with weight above a specific threshold.
- What constitutes a good choice of a term score?

TF/IDF [S83]

- Term frequency (tf) inverse document frequency (idf).
- Widely used in traditional IR approaches.
- The tf/idf value of a ‘term’ in a document:
  - Log (tf+1) * log idf where
    - tf : # of times ‘term’ appears in a document d
    - idf : number of documents / number of documents containing ‘term’
  - Intuitively: rare ‘terms’ are more important

TF/IDF

- Varying semantics of ‘term’
  - Words in a field
    - ‘AT&T Corporation’ -> ‘AT&T’ , ‘Corporation’
  - Qgrams (sequence of q-characters in a field)
    - {‘AT&’,’T&T’,’&T ‘, ‘T C’,’ Co’,’orp’,’rpo’,’por’,’ora’,’rat’,’ati’,’tio’,’ion’} 3-grams
- For each ‘term’ in a field compute its corresponding tfidf score using the field as a document and the set of field values as the document collection.

Probabilistic analog (from FS model)

- Ps(j) : probability for j in set S
- γ^j : event that values of corresponding fields are j in a random draw from sets A and B
- $m(\gamma^j) = P(\gamma^j|M) = P_{A\cap B}(j)$
- $u(\gamma^j) = P(\gamma^j|U) = P_A(j)P_B(j)$

- Assume $P_A(j) = P_B(j) = P_{A\cap B}(j)$
  - Provide more weight to agreement on rare terms and less weight to common terms
- IDF measure related to Fellegi-Sunter probabilistic notion:
  - $\mathrm{Log}(m(\gamma^{str})/u(\gamma^{str})) = \log(P_{A\cap B}(str)/P_A(str)P_B(str)) = \log(1/P_A(str)) = \mathrm{IDF}(str)$

Cosine similarity

- Each field value transformed via tfidf weighting to a (sparse) vector of high dimensionality d.
- Let a,b two field values and S_a, S_b the set of terms for each. For w in S_a (S_b), denote $W(w,S_a)$ ($W(b,S_b)$) its tfidf score.
- For two such values:
  - Cosine(a,b) = $\sum_{z \in Sa \cap Sb} W(z,Sa)W(z,Sb)$

Cosine similarity

- Suitable to assess closeness of
  - ‘AT&T Corporation’, ‘AT&T Corp’ or ‘AT&T Inc’
    - Low weights for ‘Corporation’,’Corp’,’Inc’
    - Higher weight for ‘AT&T’
    - Overall Cosine(‘AT&T Corp’,’AT&T Inc’) should be high
  - Via q-grams may capture small typing mistakes
    - ‘Jaccard’ vs ‘Jacard’ -> {‘Jac’,’acc’,’cca’,’car’,’ard’} vs {‘Jac’,’aca’,’car’,’ard’}
    - Common terms ‘Jac’, ‘car’, ‘ard’ would be enough to result in high value of Cosine(‘Jaccard’,’Jacard’).

Hybrids [CRF03]

- Let S = $\{a_1,\ldots,a_K\}$, T = $\{b_1,\ldots b_L\}$ sets of terms:
- Sim(S,T) = $\frac{1}{K}\sum_{i=1}^{K}\max_{j=1}^{L} sim'(a_i, b_j)$
- Sim'() some other similarity function
- $C(t,S,T) = \{w \in S \text{ s.t } \exists\, v \in T, sim'(w,v) > t\}$
- $D(w,T) = \max_{v \in T} sim'(w,v),\ w \in C(t,S,T)$
  - sTFIDF = $\sum_{w \in C(t,S,T)} W(w,S) * W(w,T) * D(w,T)$

Fuzzy Match Similarity [CGGM03]

- Sets of terms S, T
- Main idea: cost of transforming S to T, tc(S,T).
- Transformation operations like edit distance.
  - Replacement cost: edit(s,t)*W(s,S)
  - Insertion cost: c_{ins} W(s,S) (c_{ins} between 0,1)
  - Deletion cost: W(s,S)
- Computed by DP like edit()
- Generalized for multiple sets of terms

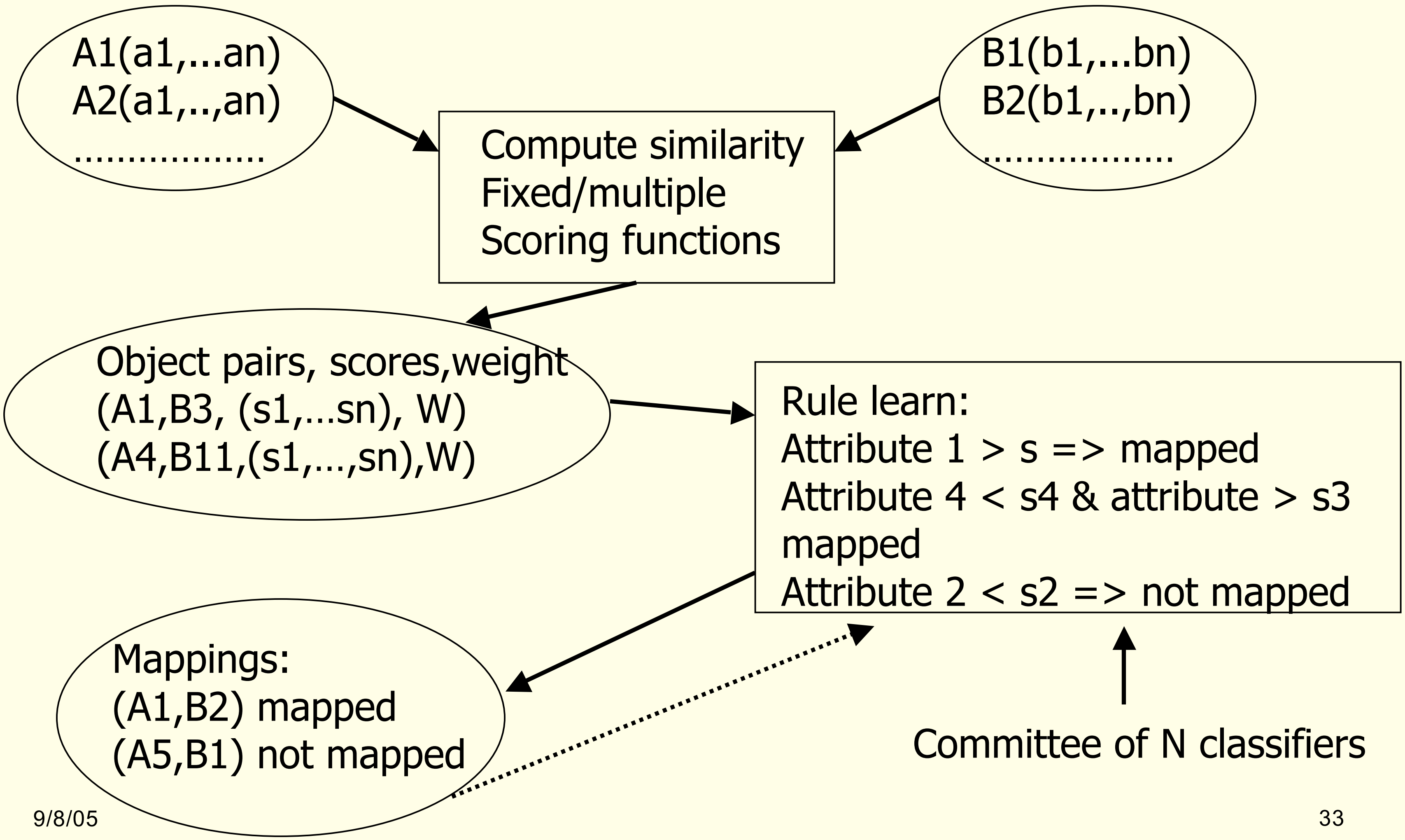
Fuzzy Match Similarity

- Example
  - ‘Beoing Corporation’,’Boeing Company’
  - S = {‘Beoing’,’Corporation}, T = {‘Boeing’,Company’}
  - tc(S,T) = 0.97 (unit weights for terms)
    - edit(‘Beoing’,’Boeing’) = 2/6 (normalized)
    - edit(‘Corporation’,Company’) = 7/11

Fuzzy Match Similarity

- W(S) = sum of W(s,S) for all s $\in$S
- fms = 1-min((tc(S,T)/W(S),1)
- Approximating fms:
  - For s $\in$ S let QG(s) set of qgrams of s
  - d= (1-1/q)
  - fms^{apx} = $\frac{1}{W(S)}\sum_{s\in S} W(s,S) * \max_{t\in T}(\frac{2}{q} sim_{mh}(QG(s),QG(t)) + d)$
  - For suitable δ, ε and size of min hash signature
    - E(fms^{apx}(S,T)) $\geq$ fms(S,T)
    - P(fms^{apx}(S,T) $\leq$ (1-δ)fms(S,T)) $\leq \varepsilon$

Learning based approaches [TKM01]



Method for creating committees

- Randomizing model selection
  - Perturbations on parameters of model learned
- Partitioning training data
  - Disjoint and overlapping partitioning
- Partitioning attributes
  - Partitioning attribute set in various ways

Voting theory application [GKMS04]

- Relations R with n attributes.
- In principle can apply a different similarity function for each pair of attributes into consideration.
- N orders of the relation tuples, ranked by a similarity score to a query.

Voting Theory

Tuple id	custname	address	location
T1	John smith	800 Mountain Av springfield	5,5
T2	Josh Smith	100 Mount Av Springfield	8,8
T3	Nicolas Smith	800 spring Av Union	11,11
T4	Joseph Smith	555 Mt. Road Springfield	9,9
T5	Jack Smith	100 Springhill lake Park	6,6

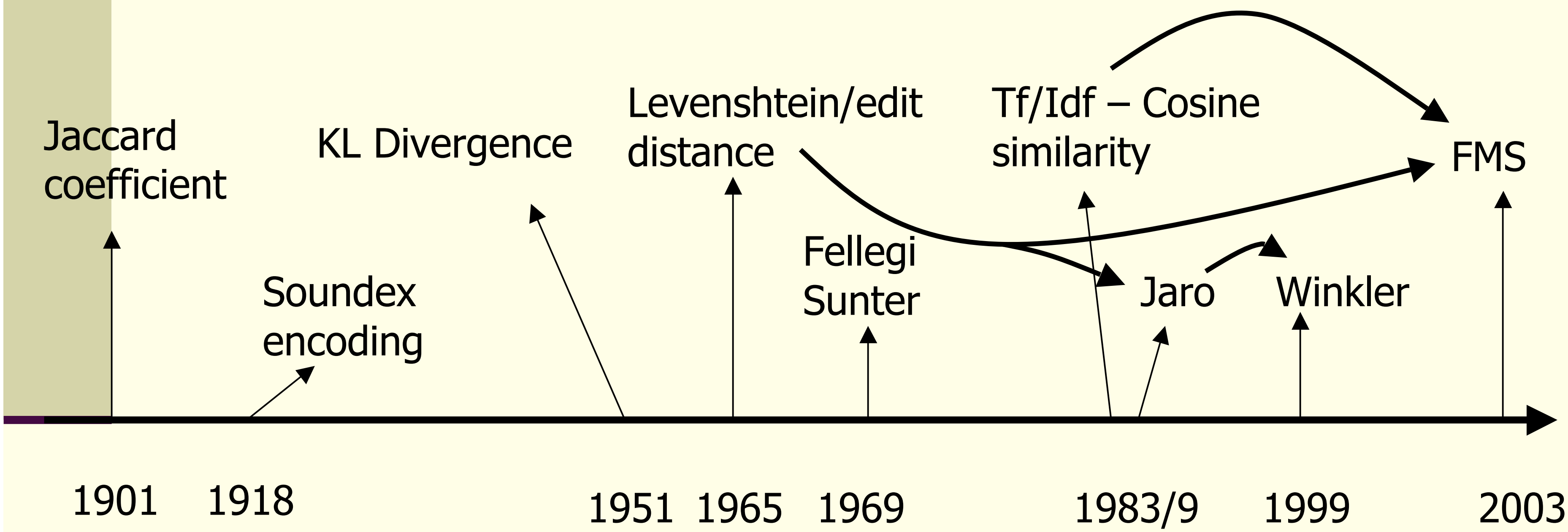
Query: John smith 100 Mount Rd. Springfield 5.1,5.1

custname	address	location
T1 (1.0)	T2 (0.95)	T1 (0.95)
T2 (0.8)	T1 (0.8)	T5 (0.9)
T5 (0.7)	T4 (0.75)	T2 (0.7)
T4 (0.6)	T3 (0.3)	T4 (0.6)
T3 (0.4)	T5 (0.1)	T3 (0.3)

Voting theory application

- Merge rankings to obtain a consensus
- Foot-rule distance
  - Let S,T orderings of the same domain D
  - S(i) (T(i)) the order position of the i-th element of D in S (T)
  - F(S,T) = $\sum_{i \in D} | S(i) - T(i) |$
  - Generalized to distance between S and T1,..Tn
    - F(S,T1,..Tn) = $\sum_{j=1}^{n} F(S, Ti)$

Historical timeline



Outline

- Part I: Motivation (15 min)
- Part II: Similarity metrics (35 min)
- Part III: Efficient algorithms (40 min)
  - Use traditional join methods
  - Extend traditional join methods
  - Open problems

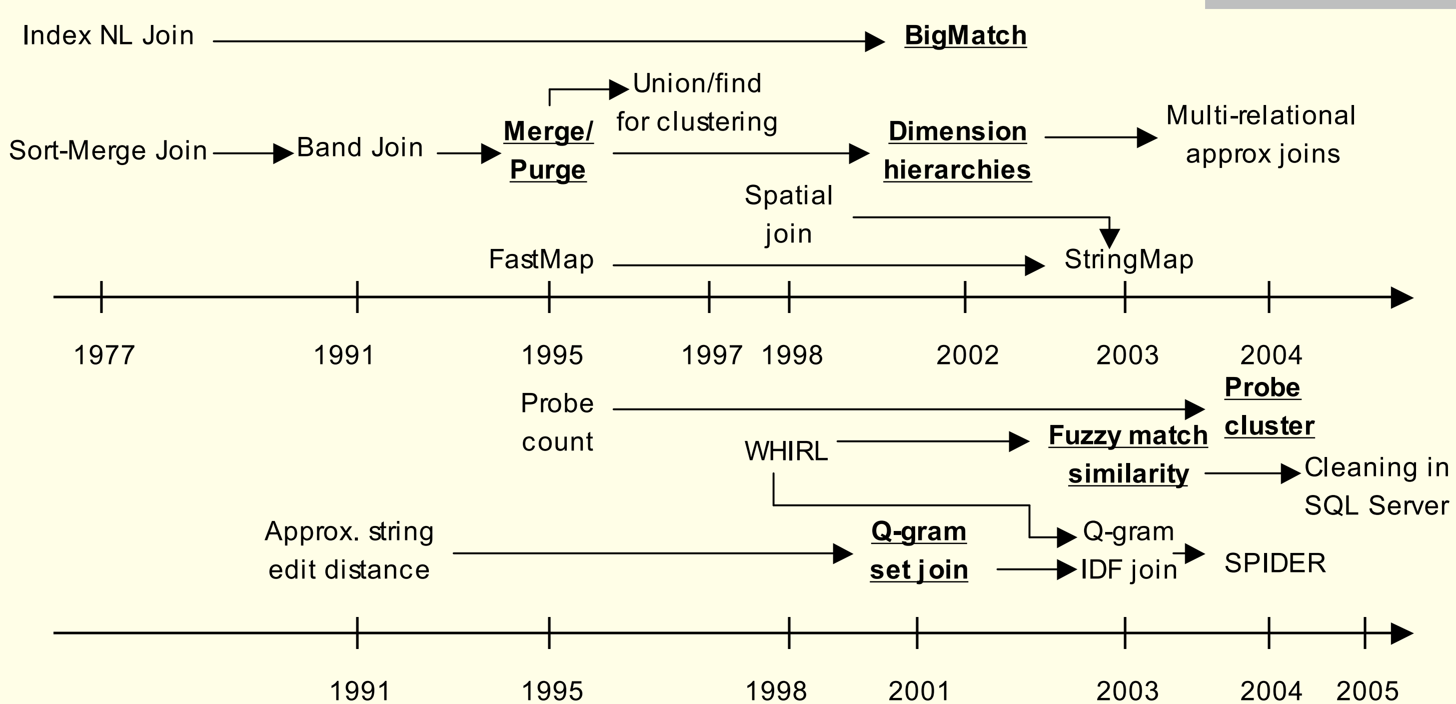
Approximate Joins: Baseline + Goal

- An **approximate join** of $R_1(A_1, \ldots, A_n)$ and $R_2(B_1, \ldots, B_m)$ is
  - A subset of the cartesian product of R_1 and R_2
  - "Matching" specified attributes $A_{i1}, \ldots, A_{ik}$ with $B_{i1}, \ldots, B_{ik}$
  - Labeled with a similarity score $> t > 0$

- Naïve method: for each record pair, compute similarity score
  - I/O and CPU intensive, not scalable to millions of records

- Goal: reduce $O(n^2)$ cost to $O(n*w)$, where $w << n$
  - Reduce number of pairs on which similarity is computed
  - Take advantage of efficient relational join methods

Historical Timelines



Sorted Neighborhood Method [HS95]

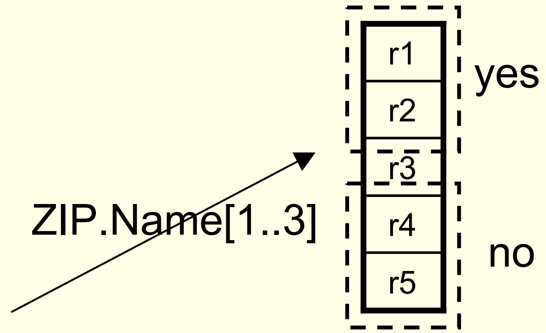
- Goal: bring matching records close to each other in linear list
- Background: duplicate elimination [DB83], band join [DNS91]
- Methodology: domain-specific, arbitrary similarity
  - Compute discriminating key per record, sort records
  - Slide fixed size window through sorted list, match in window
  - Use OPS5 rules (equational theory) to determine match
  - Multiple passes with small windows, based on distinct keys
- Lesson: multiple “cheap” passes faster than an “expensive” one

Sorted Neighborhood Method [HS95]

- Goal: bring matching records close to each other in linear list

- Example:

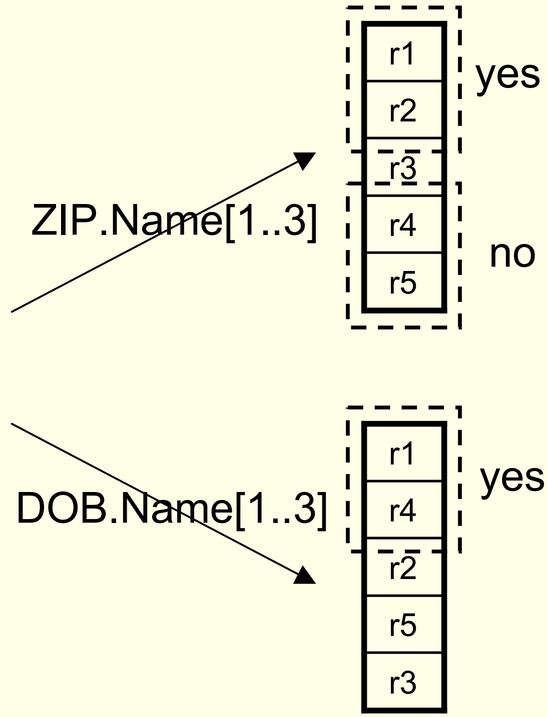
ID	Name	SS	DOB	ZIP
r1	Smith, John	123-45	1960/08/24	07932
r2	Smyth, Jon	123-45	1961/08/24	07932
r3	Smith, John	312-54	1995/07/25	98301
r4	Smith, J.	723-45	1960/08/24	98346
r5	Smith, J.	456-78	1975/12/11	98346



Sorted Neighborhood Method [HS95]

- Goal: bring matching records close to each other in linear list
- Example:

ID	Name	SS	DOB	ZIP
r1	Smith, John	123-45	1960/08/24	07932
r2	Smyth, Jon	123-45	1961/08/24	07932
r3	Smith, John	312-54	1995/07/25	98301
r4	Smith, J.	723-45	1960/08/24	98346
r5	Smith, J.	456-78	1975/12/11	98346



- Blocking is a special case

BigMatch [Y02]

- Goal: block/index matching records, based on multiple keys
- Background: indexed nested loop join [BE77]
- Methodology: domain-specific, Jaro-Winkler similarity
  - Store smaller table (100M) in main memory (4GB)
  - Create indexes for each set of grouping/blocking criteria
  - Scan larger table (4B), repeatedly index smaller table
  - Avoids multiple matches of the same pair
- Lesson: traditional join technique applicable for approximate join

BigMatch [Y02]

- Goal: block/index matching records, based on multiple keys
- Example:

inner table

record from outer table

SS.Name[1..2]

yes

no

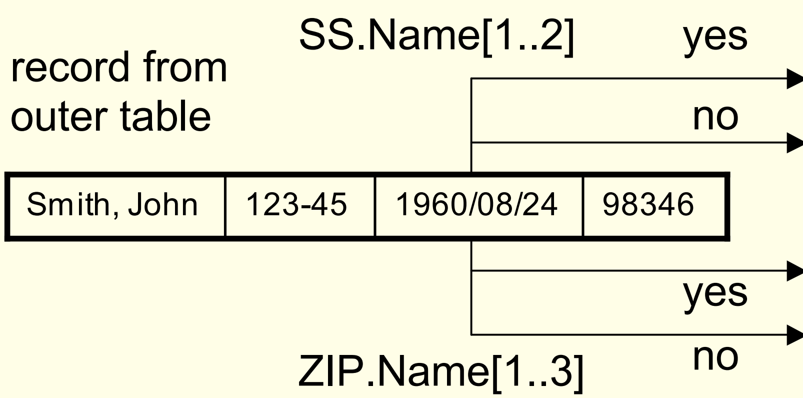
Smith, John	123-45	1960/08/24	98346

ID	Name	SS	DOB	ZIP
r1	Smith, John	123-45	1960/08/24	07932
r2	Smyth, Jon	123-45	1961/08/24	07932
r3	Smith, John	312-54	1995/07/25	98301
r4	Smith, J.	723-45	1960/08/24	98346
r5	Smith, J.	456-78	1975/12/11	98346

BigMatch [Y02]

- Goal: block/index matching records, based on multiple keys

- Example:



inner table

ID	Name	SS	DOB	ZIP
r1	Smith, John	123-45	1960/08/24	07932
r2	Smyth, Jon	123-45	1961/08/24	07932
r3	Smith, John	312-54	1995/07/25	98301
r4	Smith, J.	723-45	1960/08/24	98346
r5	Smith, J.	456-78	1975/12/11	98346

- Avoids multiple matches of the same pair

Use Dimension Hierarchies [ACG02]

- Goal: exploit dimension hierarchies for duplicate elimination

- Background: clustering categorical data [GKR98]

- Methodology: domain-independent, structure+text similarity
  - Use hierarchical grouping, instead of sorting, to focus search
  - “Structural” similarity based on overlap of children sets
  - Textual similarity based on weighted token set containment
  - Top-down processing of dimension hierarchy for efficiency

- Lesson: useful to consider structure in addition to content

Use Dimension Hierarchies [ACG02]

- Goal: exploit dimension hierarchies for duplicate elimination
- Example:

AI	Address	CI
a1	10 Mountain Avenue	c1
a2	250 McCarter	c2
a3	250 McCarter Hwy	c3
a4	10 Mountain	c4
a5	10 Mountain Street	c5

CI	City	SI
c1	Summit	s1
c2	Newark	s2
c3	Newark	s3
c4	Summit	s4
c5	Summitt	s5

SI	State	YI
s1	NJ	y1
s2	New Jersey	y1
s3	NJ	y2
s4	New Jersey	y2
s5	NJ	y3

YI	Country
y1	USA
y2	United States
y3	US

Use Dimension Hierarchies [ACG02]

- Goal: exploit dimension hierarchies for duplicate elimination

- Example:

AI	Address	CI
a1	10 Mountain Avenue	c1
a2	250 McCarter	c2
a3	250 McCarter Hwy	c3
a4	10 Mountain	c4
a5	10 Mountain Street	c5

CI	City	SI
c1	Summit	s1
c2	Newark	s2
c3	Newark	s3
c4	Summit	s4
c5	Summitt	s5

SI	State	YI
s1	NJ	y1
s2	New Jersey	y1
s3	NJ	y2
s4	New Jersey	y2
s5	NJ	y1

YI	Country
y1	USA
y2	United States
~~y3~~	~~US~~

- Textual similarity

Use Dimension Hierarchies [ACG02]

- Goal: exploit dimension hierarchies for duplicate elimination

- Example:

AI	Address	CI	CI	City	SI	SI	State	YI	YI	Country
a1	10 Mountain Avenue	c1	c1	Summit	s1	s1	NJ	y1	y1	USA
a2	250 McCarter	c2	c2	Newark	s2	s2	New Jersey	y1	~~y2~~	~~United States~~
a3	250 McCarter Hwy	c3	c3	Newark	s3	s3	NJ	y1	~~y3~~	~~US~~
a4	10 Mountain	c4	c4	Summit	s4	s4	New Jersey	y1		
a5	10 Mountain Street	c5	c5	Summitt	s5	s5	NJ	y1		

- Structural similarity

Use Dimension Hierarchies [ACG02]

- Goal: exploit dimension hierarchies for duplicate elimination

- Example:

AI	Address	CI
a1	10 Mountain Avenue	c1
a2	250 McCarter	c2
a3	250 McCarter Hwy	c3
a4	10 Mountain	c4
a5	10 Mountain Street	c5

CI	City	SI
c1	Summit	s1
c2	Newark	s2
c3	Newark	s1
c4	Summit	s2
c5	Summitt	s1

SI	State	YI
s1	NJ	y1
s2	New Jersey	y1
~~s3~~	~~NJ~~	~~y1~~
~~s4~~	~~New Jersey~~	~~y1~~
~~s5~~	~~NJ~~	~~y1~~

YI	Country
y1	USA
~~y2~~	~~United States~~
~~y3~~	~~US~~

Use Dimension Hierarchies [ACG02]

- Goal: exploit dimension hierarchies for duplicate elimination

- Example:

AI	Address	CI	CI	City	SI	SI	State	YI	YI	Country
a1	10 Mountain Avenue	c1	c1	Summit	s1	s1	NJ	y1	y1	USA
a2	250 McCarter	c2	c2	Newark	s1	~~s2~~	~~New Jersey~~	~~y1~~	~~y2~~	~~United States~~
a3	250 McCarter Hwy	c3	c3	Newark	s1	~~s3~~	~~NJ~~	~~y1~~	~~y3~~	~~US~~
a4	10 Mountain	c4	c4	Summit	s1	~~s4~~	~~New Jersey~~	~~y1~~		
a5	10 Mountain Street	c5	c5	Summitt	s1	~~s5~~	~~NJ~~	~~y1~~		

Use Dimension Hierarchies [ACG02]

- Goal: exploit dimension hierarchies for duplicate elimination

- Example:

AI	Address	CI
a1	10 Mountain Avenue	c1
a2	250 McCarter	c2
a3	250 McCarter Hwy	c2
a4	10 Mountain	c1
a5	10 Mountain Street	c1

CI	City	SI
c1	Summit	s1
c2	Newark	s1
~~c3~~	~~Newark~~	~~s1~~
~~c4~~	~~Summit~~	~~s1~~
~~c5~~	~~Summitt~~	~~s1~~

SI	State	YI
s1	NJ	y1
~~s2~~	~~New Jersey~~	~~y1~~
~~s3~~	~~NJ~~	~~y1~~
~~s4~~	~~New Jersey~~	~~y1~~
~~s5~~	~~NJ~~	~~y1~~

YI	Country
y1	USA
~~y2~~	~~United States~~
~~y3~~	~~US~~

Use Dimension Hierarchies [ACG02]

- Goal: exploit dimension hierarchies for duplicate elimination

- Example:

AI	Address	CI
a1	10 Mountain Avenue	c1
~~a2~~	~~250 McCarter~~	c2
a3	250 McCarter Hwy	c2
~~a4~~	~~10 Mountain~~	c1
~~a5~~	~~10 Mountain Street~~	c1

CI	City	SI
c1	Summit	s1
c2	Newark	s1
~~c3~~	~~Newark~~	~~s1~~
~~c4~~	~~Summit~~	~~s1~~
~~c5~~	~~Summitt~~	~~s1~~

SI	State	YI
s1	NJ	y1
~~s2~~	~~New Jersey~~	~~y1~~
~~s3~~	~~NJ~~	~~y1~~
~~s4~~	~~New Jersey~~	~~y1~~
~~s5~~	~~NJ~~	~~y1~~

YI	Country
y1	USA
~~y2~~	~~United States~~
~~y3~~	~~US~~

Q-gram Set Join [GIJ+01]

- Goal: compute thresholded edit distance join on string attributes

- Background: combinatorial pattern matching [JU91]

- Methodology: domain-independent, edit distance similarity
  - Extract set of all overlapping q-grams Q(s) from string s
  - $ED(s_1,s_2) \le d \rightarrow |Q(s_1) \cap Q(s_2)| \ge \max(|s_1|,|s_2|) - (d-1)*q - 1$
  - Cheap filters (length, count, position) to prune non-matches
  - Pure SQL solution: cost-based join methods

- Lesson: reduce approximate join to aggregated set intersection

Q-gram Set Join [GIJ+01]

- Goal: compute thresholded edit distance join on string attributes

- Example:

ID	Name
r1	Srivastava
r2	Shrivastava
r3	Shrivastav

Q-gram Set Join [GIJ+01]

- Goal: compute thresholded edit distance join on string attributes

- Example:

ID	Name	3-grams
r1	Srivastava	##s, #sr, sri, riv, iva, vas, ast, sta, tav, ava, va$, a$$
r2	Shrivastava	##s, #sh, shr, hri, riv, iva, vas, ast, sta, tav, ava, va$, a$$
r3	Shrivastav	

  - $ED(s_1,s_2) \leq d \rightarrow |Q(s_1) \cap Q(s_2)| \geq \max(|s_1|,|s_2|) - (d-1)*q - 1$
  - $ED(r1, r2) = 1$, $|Q(r1) \cap Q(r2)| = 10$

Q-gram Set Join [GIJ+01]

- Goal: compute thresholded edit distance join on string attributes
- Example:

ID	Name	3-grams
r1	Srivastava	##s, #sr, sri, riv, iva, vas, ast, sta, tav, ava, va$, a$$
r2	Shrivastava	
r3	Shrivastav	##s, #sh, shr, hri, riv, iva, vas, ast, sta, tav, av$, v$$

  - $ED(s_1,s_2) \leq d \rightarrow |Q(s_1) \cap Q(s_2)| \geq \max(|s_1|,|s_2|) - (d-1)*q - 1$
  - $ED(r1, r2) = 2$, $|Q(r1) \cap Q(r2)| = 7$

Q-gram Set Join [GIJ+01]

- Goal: compute thresholded edit distance join on string attributes

- Example:

ID	Name
r1	Srivastava
r2	Shrivastava
r3	Shrivastav

Q

ID	Qg
r1	##s
r1	#sr
r1	sri
r1	riv
r1	iva
r1	vas
r1	ast
r1	sta
r1	tav
r1	ava
r1	va$
r1	a$$

ID	Qg
r3	##s
r3	#sh
r3	shr
r3	hri
r3	riv
r3	iva
r3	vas
r3	ast
r3	sta
r3	tav
r3	av$
r3	v$$

Q-gram Set Join [GIJ+01]

- Goal: compute thresholded edit distance join on string attributes
- Example:

ID	Name
r1	Srivastava
r2	Shrivastava
r3	Shrivastav

```
SELECT Q1.ID, Q2.ID
FROM Q AS Q1, Q AS Q2
WHERE Q1.Qg = Q2.Qg
GROUP BY Q1.ID, Q2.ID
HAVING COUNT(*) > T
```

Q

ID	Qg
r1	##s
r1	#sr
r1	sri
r1	riv
r1	iva
r1	vas
r1	ast
r1	sta
r1	tav
r1	ava
r1	va$
r1	a$$

ID	Qg
r3	##s
r3	#sh
r3	shr
r3	hri
r3	riv
r3	iva
r3	vas
r3	ast
r3	sta
r3	tav
r3	av$
r3	v$$

Fuzzy Match Similarity [CGGM03]

- Goal: identify K “closest” reference records in on-line setting
- Background: IDF weighted cosine similarity, WHIRL [C98]
- Methodology: domain-independent, cosine+ED similarity
  - Similarity metric based on IDF weighted token edit distance
  - Approximate similarity metric using Jaccard on q-gram sets
  - Small error tolerant index table, sharing of minhash q-grams
  - Optimistic short circuiting exploits large token IDF weights
- Lesson: IDF weighting useful to capture erroneous tokens

Fuzzy Match Similarity [CGGM03]

- Goal: identify K “closest” reference records in on-line setting
- Example:

reference table

ID	OrgName	City	State	ZIP
r1	Boeing Company	Seattle	WA	98004
r2	Bon Corporation	Seattle	WA	98014
r3	Companions	Seattle	WA	98024

best ED match

input record

Beoing Corporation	Seattle	WA	98004

Fuzzy Match Similarity [CGGM03]

- Goal: identify K “closest” reference records in on-line setting

- Example:

reference table

best FMS match

ID	OrgName	City	State	ZIP
r1	Boeing Company	Seattle	WA	98004
r2	Bon Corporation	Seattle	WA	98014
r3	Companions	Seattle	WA	98024

input record

Beoing Corporation	Seattle	WA	98004

Fuzzy Match Similarity [CGGM03]

- Goal: identify K "closest" reference records in on-line setting
- Example:

reference table

ID	OrgName	City	State	ZIP
r1	Boeing Company	Seattle	WA	98004
r2	Bon Corporation	Seattle	WA	98014
r3	Companions	Seattle	WA	98024

input record

Beoing Corporation	Seattle	WA	98004

[eoi, ing] [orp, ati] [sea, ttl] [wa] [980, 004]

all minhash q-grams

ETI table

Qg	MHC	Col	Freq	TIDList
ing	2	1	1	{r1}
orp	1	1	1	{r2}
sea	1	2	3	{r1, r2, r3}
004	2	4	1	{r1}

Fuzzy Match Similarity [CGGM03]

- Goal: identify K "closest" reference records in on-line setting

- Example:

reference table

ID	OrgName	City	State	ZIP
r1	Boeing Company	Seattle	WA	98004
r2	Bon Corporation	Seattle	WA	98014
r3	Companions	Seattle	WA	98024

input record

Beoing Corporation	Seattle	WA	98004

[eoi, ing] [orp, ati] [sea, ttl] [wa] [980, 004]

ETI table

Qg	MHC	Col	Freq	TIDList
ing	2	1	1	{r1}
orp	1	1	1	{r2}
sea	1	2	3	{r1, r2, r3}
004	2	4	1	{r1}

optimistic short circuiting

Probe-Cluster: Set Joins [SK04]

- Goal: generic algorithm for set join based on similarity predicate
- Background: IR and probe count using inverted index [TF95]
- Methodology: domain-independent, arbitrary set similarity
  - Jaccard similarity, cosine similarity, and their variants
  - Build inverted lists on individual set elements
  - Skewness → process lists in increasing size order
  - Sort lists in decreasing order of record sizes
- Lesson: IR query optimizations useful for approximate joins

Probe-Cluster: Set Joins [SK04]

- Goal: generic algorithm for set join based on similarity predicate

- Example:

ID	SVA
r1	{##s, #sr, sri, riv, iva, vas, ast, sta, tav, ava, va$, a$$}
r2	{##s, #sh, shr, hri, riv, iva, vas, ast, sta, tav, ava, va$, a$$}
r3	{##s, #sh, shr, hri, riv, iva, vas, ast, sta, tav, av$, v$$}

Inverted index

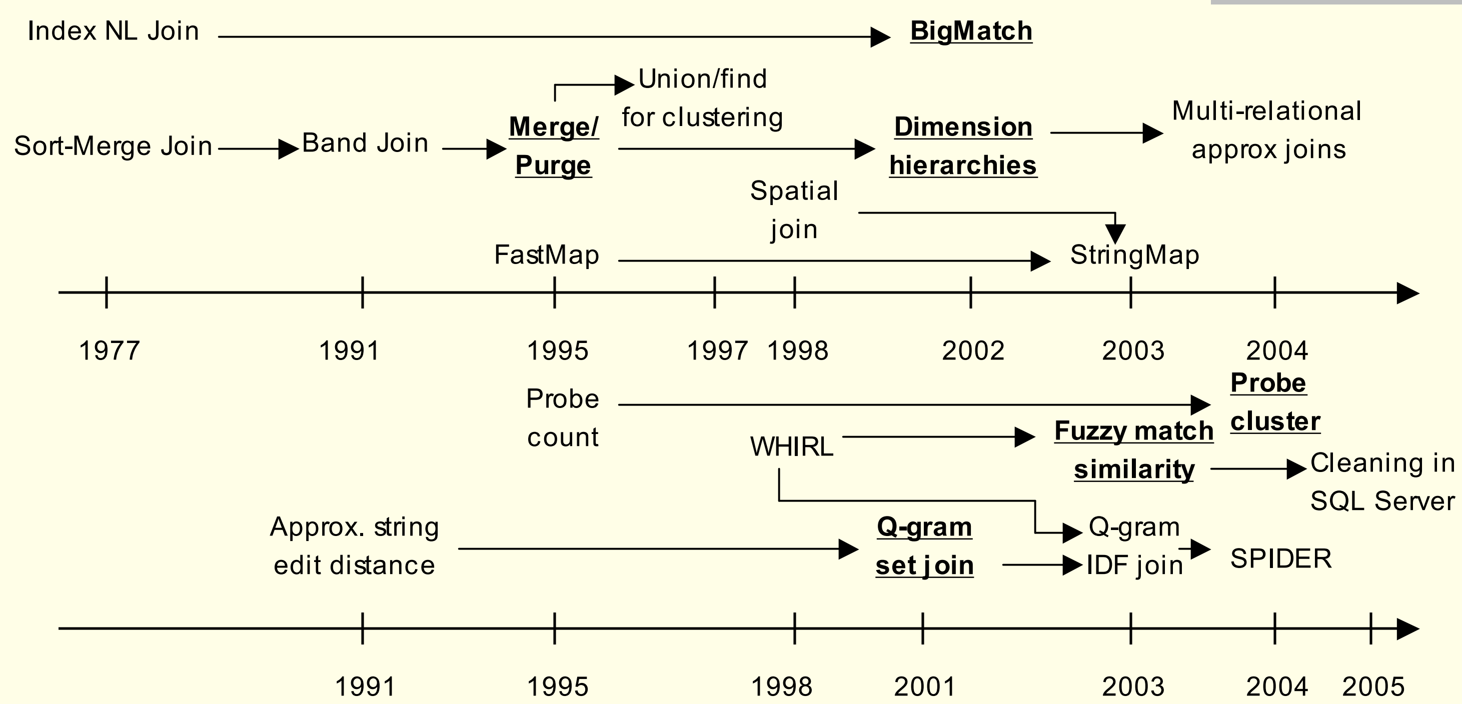
SE	IDs
##s	r1, r2, r3
#sr	r1
#sh	r2, r3
sri	r1
shr	r2, r3
hri	r2, r3
riv	r1, r2, r3
…	…
tav	r1, r2, r3
ava	r1, r2
…	…
v$$	r3

Duplicate Elimination

- Duplicate elimination = approximate joins, clustering, merging

- Clustering techniques for duplicate elimination
  - Transitive closure (single linkage) [HS95]
  - Priority queues, union/find [ME97]
  - Canopy method [MNU00]
  - Compact sets, sparse neighborhood [CGM05]

Historical Timelines



Open Problem: Inside or Outside?

- Issue: optimizable processing in a relational database

- Background
  - Declarative data cleaning in AJAX [GFS+01]
  - Q-gram based metrics [GIJ+01,GIKS03,KMS04]
  - Compact sets, sparse neighborhood [CGM05]

- Goal: express arbitrary approximate joins (not just UDF) in SQL

Open Problem: Multi-table Joins

- Issue: information in auxiliary tables can aid matching
- Background
  - Hierarchical models [ACG02]
  - Iterative matching [BG04]
  - Graphical models [KMC05]
- Goal: efficient multi-table approximate joins

Open Problem: Benchmarking

- Issue: many algorithms and similarity measures, no benchmarks
- Background
  - Comparing quality of different similarity measures [CRF03]
- Goal: develop standard benchmarks (queries, data generation)

Conclusions

- Approximate joins are critical when data quality is poor
  - Similarity metrics
  - Efficient sub-quadratic algorithms

- Wealth of challenging technical problems
  - Sophisticated similarity metrics, massive data sets
  - Important to work with real datasets

References

- [ACG02] Rohit Ananthakrishna, Surajit Chaudhuri, Venkatesh Ganti: Eliminating Fuzzy Duplicates in Data Warehouses. VLDB 2002: 586-597
- [BD83] Dina Bitton, David J. DeWitt: Duplicate Record Elimination in Large Data Files. ACM Trans. Database Syst. 8(2): 255-265 (1983)
- [BE77] Mike W. Blasgen, Kapali P. Eswaran: Storage and Access in Relational Data Bases. IBM Systems Journal 16(4): 362-377 (1977)
- [BG04] Indrajit Bhattacharya, Lise Getoor: Iterative record linkage for cleaning and integration. DMKD 2004: 11-18
- [C98] William W. Cohen: Integration of Heterogeneous Databases Without Common Domains Using Queries Based on Textual Similarity. SIGMOD Conference 1998: 201-212
- [C00] William W. Cohen: Data integration using similarity joins and a word-based information representation language. ACM Trans. Inf. Syst. 18(3): 288-321 (2000)
- [CCZ02] Peter Christen, Tim Churches, Xi Zhu: Probabilistic name and address cleaning and standardization. Australasian Data Mining Workshop 2002.
- [CGGM04] Surajit Chaudhuri, Kris Ganjam, Venkatesh Ganti, Rajeev Motwani: Robust and Efficient Fuzzy Match for Online Data Cleaning. SIGMOD Conference 2003: 313-324
- [CGG+05] Surajit Chaudhuri, Kris Ganjam, Venkatesh Ganti, Rahul Kapoor, Vivek R. Narasayya, Theo Vassilakis: Data cleaning in microsoft SQL server 2005. SIGMOD Conference 2005: 918-920
- [CGM05] Surajit Chaudhuri, Venkatesh Ganti, Rajeev Motwani: Robust Identification of Fuzzy Duplicates. ICDE 2005: 865-876
- [CRF03] William W. Cohen, Pradeep Ravikumar, Stephen E. Fienberg: A Comparison of String Distance Metrics for Name-Matching Tasks. IIWeb 2003: 73-78

References

- [DJ03] Tamraparni Dasu, Theodore Johnson: Exploratory Data Mining and Data Cleaning John Wiley 2003
- [DNS91] David J. DeWitt, Jeffrey F. Naughton, Donovan A. Schneider: An Evaluation of Non-Equijoin Algorithms. VLDB 1991: 443-452
- [DWI02] Data Warehousing Institute report 2002
- [E00] Larry English: Plain English on Data Quality: Information Quality Management: The Next Frontier. DM Review Magazine: April 2000. http://www.dmreview.com/article_sub.cfm?articleId=2073
- [FL95] Christos Faloutsos, King-Ip Lin: FastMap: A Fast Algorithm for Indexing, Data-Mining and Visualization of Traditional and Multimedia Datasets. SIGMOD Conference 1995: 163-174
- [FS69] I. Fellegi, A. Sunter: A theory of record linkage. Journal of the American Statistical Association, Vol 64. No 328, 1969
- [G98] D. Gusfield: Algorithms on strings, trees and sequences. Cambridge university press 1998
- [GFS+01] Helena Galhardas, Daniela Florescu, Dennis Shasha, Eric Simon, Cristian-Augustin Saita: Declarative Data Cleaning: Language, Model, and Algorithms. VLDB 2001: 371-380
- [GIJ+01] Luis Gravano, Panagiotis G. Ipeirotis, H. V. Jagadish, Nick Koudas, S. Muthukrishnan, Divesh Srivastava: Approximate String Joins in a Database (Almost) for Free. VLDB 2001: 491-500
- [GIKS03] Luis Gravano, Panagiotis G. Ipeirotis, Nick Koudas, Divesh Srivastava: Text joins in an RDBMS for web data integration. WWW 2003: 90-101
- [GKMS04] S. Guha, N. Koudas, A. Marathe, D. Srivastava : Merging the results of approximate match operations. VLDB 2004.
- [GKR98] David Gibson, Jon M. Kleinberg, Prabhakar Raghavan: Clustering Categorical Data: An Approach Based on Dynamical Systems. VLDB 1998: 311-322

References

- [HS95] Mauricio A. Hernández, Salvatore J. Stolfo: The Merge/Purge Problem for Large Databases. SIGMOD Conference 1995: 127-138
- [HS98] Gísli R. Hjaltason, Hanan Samet: Incremental Distance Join Algorithms for Spatial Databases. SIGMOD Conference 1998: 237-248
- [J89] M. A. Jaro: Advances in record linkage methodology as applied to matching the 1985 census of Tampa, Florida. Journal of the American Statistical Association 84: 414-420.
- [JLM03] Liang Jin, Chen Li, Sharad Mehrotra: Efficient Record Linkage in Large Data Sets. DASFAA 2003
- [JU91] Petteri Jokinen, Esko Ukkonen: Two Algorithms for Approximate String Matching in Static Texts. MFCS 1991: 240-248
- [KL51] S. Kullback, R. Liebler : On information and sufficiency. The annals of mathematical statistics 22(1): 79-86. 1959.
- [KMC05] Dmitri V. Kalashnikov, Sharad Mehrotra, Zhaoqi Chen: Exploiting Relationships for Domain-Independent Data Cleaning. SDM 2005
- [KMS04] Nick Koudas, Amit Marathe, Divesh Srivastava: Flexible String Matching Against Large Databases in Practice. VLDB 2004: 1078-1086
- [KMS05] Nick Koudas, Amit Marathe, Divesh Srivastava: SPIDER: flexible matching in databases. SIGMOD Conference 2005: 876-878
- [LLL00] Mong-Li Lee, Tok Wang Ling, Wai Lup Low: IntelliClean: a knowledge-based intelligent data cleaner. KDD 2000: 290-294
- [ME96] Alvaro E. Monge, Charles Elkan: The Field Matching Problem: Algorithms and Applications. KDD 1996: 267-270

References

- [ME97] Alvaro E. Monge, Charles Elkan: An Efficient Domain-Independent Algorithm for Detecting Approximately Duplicate Database Records. DMKD 1997
- [RY97] E. Ristad, P. Yianilos : Learning string edit distance. IEEE Pattern analysis and machine intelligence 1998.
- [S83] Gerry Salton : Introduction to modern information retrieval. McGraw Hill 1987.
- [SK04] Sunita Sarawagi, Alok Kirpal: Efficient set joins on similarity predicates. SIGMOD Conference 2004: 743-754
- [TF95] Howard R. Turtle, James Flood: Query Evaluation: Strategies and Optimizations. Inf. Process. Manage. 31(6): 831-850 (1995)
- [TKF01] S. Tejada, C. Knoblock, S. Minton : Learning object identification rules for information integration. Information Systems, Vol 26, No 8, 607-633, 2001.
- [W94] William E. Winkler: Advanced methods for record linkage. Proceedings of the section on survey research methods, American Statistical Association 1994: 467-472
- [W99] William E. Winkler: The state of record linkage and current research problems. IRS publication R99/04 (http://www.census.gov/srd/www/byname.html)
- [Y02] William E. Yancey: BigMatch: A program for extracting probable matches from a large file for record linkage. RRC 2002-01. Statistical Research Division, U.S. Bureau of the Census.