On Data Dependencies in Dataspaces

Shaoxu Song

Tsinghua University

This is a joint work with Lei Chen (HKUST) and Philip S. Yu (UIC)

sxsong@tsinghua.edu.cn

2011

Shaoxu Song sxsong@tsinghua.edu.cn

Dataspaces

- provide a co-existing system of heterogeneous data
- consider three levels of elements,
  $object : \{(attribute : value)\}$

Example

We consider a dataspace with following objects,

$t_1 : \{(\text{name} : \text{iPod}), (\text{color} : \text{red}), (\text{manu} : \text{Apple Inc.}), (\text{tel} : 567),$
$(\text{addr} : \text{InfiniteLoop}, \text{CA}), (\text{website} : \text{itunes.com})\};$
$t_2 : \{(\text{name} : \text{iPod}), (\text{color} : \text{cardinal}), (\text{prod} : \text{Apple}), (\text{tel} : 123),$
$(\text{post} : \text{InfiniteLoop}, \text{Cupert}), (\text{website} : \text{apple.com})\};$
$t_3 : \{(\text{name} : \text{iPad}), (\text{color} : \text{white}), (\text{manu} : \text{Apple Inc.}),$
$(\text{post} : \text{InfiniteLoop}), (\text{website} : \text{apple.com}), (\text{phn} : 567)\}.$

Shaoxu Song sxsong@tsinghua.edu.cn

Comparable Correspondence

Relationship between elements in heterogeneous data

- *metric operator* ‘manu $\approx_{\leq 5}$ prod’
  any two respective values of manu and prod are said comparable, e.g., Apple Inc and Apple, if their edit distance is ≤ 5.
- *matching operator* ‘color $\rightleftharpoons$ color’
  e.g., red and cardinal are said *matched* as comparable color, via users’ feedback
- often incrementally recognized in a pay-as-you-go style

A query of (manu : Apple)

- search value similar to Apple in both manu and prod
- e.g., (manu : Apple Inc.) in t_1 and (prod : Apple) in t_2

Shaoxu Song sxsong@tsinghua.edu.cn

Data Dependencies

For wider applications

- integrity constraints, schema design
- optimizing query evaluation, capturing data inconsistency, removing data duplicates

Conventional data dependencies not directly applicable to dataspaces

- often defined on the equality function
- functional dependencies (FDs), $X \rightarrow A$
- specify the constraint of equality between the values of two objects on the same attribute
- e.g., $\mathsf{manu} \rightarrow \mathsf{addr}$
- cannot address the comparable correspondence, in $(\mathsf{manu}, \mathsf{prod})$ or $(\mathsf{addr}, \mathsf{post})$

Shaoxu Song sxsong@tsinghua.edu.cn

Comparable Function

Specify constraints on comparable attributes

$\theta(\text{manu}, \text{prod}) : [\text{manu} \approx_{\leq 5} \text{manu}, \text{manu} \approx_{\leq 5} \text{prod}, \text{prod} \approx_{\leq 5} \text{prod}]$

Two objects are said comparable on $(\text{manu}, \text{prod})$ if at least one of these three comparison operators in $\theta(\text{manu}, \text{prod})$ is applicable.

- t_1, t_2 are comparable on $(\text{manu}, \text{prod})$, since edit distance of $(t_1[\text{manu}], t_2[\text{prod}])$ is ≤ 5
- t_1, t_3 are also comparable on $(\text{manu}, \text{prod})$, where $(t_1[\text{manu}], t_3[\text{manu}])$ satisfy 'manu $\approx_{\leq 5}$ manu'

t_1 : {(name : iPod), (color : red), (manu : Apple Inc.), (tel : 567), (addr : InfiniteLoop, CA), (website : itunes.com)};
t_2 : {(name : iPod), (color : cardinal), (prod : Apple), (tel : 123), (post : InfiniteLoop, Cupert), (website : apple.com)};
t_3 : {(name : iPad), (color : white), (manu : Apple Inc.), (post : InfiniteLoop), (website : apple.com), (phn : 567)}.

Shaoxu Song sxsong@tsinghua.edu.cn

Comparable Dependencies (CDs)

A general form of dependencies on comparable functions

$$\varphi_1 : \theta(\mathsf{manu}, \mathsf{prod}) \rightarrow \theta(\mathsf{addr}, \mathsf{post})$$

- if the manu or prod values of two products are comparable
- then their corresponding addr or post values should also be comparable

where

$$\theta(\mathsf{addr}, \mathsf{post}) : [\mathsf{addr} \approx_{\leq 9} \mathsf{addr}, \mathsf{addr} \approx_{\leq 9} \mathsf{post}, \mathsf{post} \approx_{\leq 9} \mathsf{post}]$$

is another comparable function

Shaoxu Song sxsong@tsinghua.edu.cn

Application Example

Query optimization

- consider an object t_1 as the query
- to query objects having values similar to $(\mathsf{manu} : \mathsf{Apple\ Inc.})$ and $(\mathsf{addr} : \mathsf{InfiniteLoop}, \mathsf{CA})$ of t_1
- search in the manu, addr attributes specified in the query,
- also search in the comparable attributes prod, post according to the comparable functions $\theta(\mathsf{manu}, \mathsf{prod})$ and $\theta(\mathsf{addr}, \mathsf{post})$
- according to φ_1, rewrite the query by using $(\mathsf{manu}, \mathsf{prod})$ only

t_1 : {(name : iPod), (color : red), (manu : Apple Inc.), (tel : 567), (addr : InfiniteLoop, CA), (website : itunes.com)};
t_2 : {(name : iPod), (color : cardinal), (prod : Apple), (tel : 123), (post : InfiniteLoop, Cupert), (website : apple.com)};
t_3 : {(name : iPad), (color : white), (manu : Apple Inc.), (post : InfiniteLoop), (website : apple.com), (phn : 567)}.

Shaoxu Song sxsong@tsinghua.edu.cn

Related Work

Metric functional dependencies (MFDs)

- $X \xrightarrow{\delta} A$
- equality operator in the left-hand-side
- similarity operator in the right-hand-side
- for violation detection
- e.g., $\mathsf{manu} \xrightarrow{2} \mathsf{addr}$

Matching dependencies (MDs)

- $[X \approx X] \rightarrow [A \rightleftharpoons A]$
- similarity operator in the left-hand-side
- matching operator in the right-hand-side
- for record matching
- e.g., $[\mathsf{addr} \approx \mathsf{addr}] \rightarrow [\mathsf{tel} \rightleftharpoons \mathsf{tel}]$

Outline

Shaoxu Song sxsong@tsinghua.edu.cn

Comparison Operator

We consider a general form of comparison operators, which include the previous operators.

Let $A_i \leftrightarrow_{ij} A_j$ denote a *comparison operator* between two attributes A_i, A_j in a dataspace $\mathcal{S}$

- equality operator $A_i = A_j$ in *functional dependencies* (FDs)
- metric operator $A_i \approx_\lambda A_j$ in *metric functional dependencies* (MFDs)
- matching operator $A_i \rightleftharpoons A_j$ in *matching dependencies* (MDs)

The comparision operator indicates true, if two values satisfy the corresponding constraint.

Shaoxu Song sxsong@tsinghua.edu.cn

Syntex

A general *comparable function*

$$\theta(A_i, A_j) : [A_i \leftrightarrow_{ii} A_i, A_i \leftrightarrow_{ij} A_j, A_j \leftrightarrow_{jj} A_j]$$

specifies a comparable constraint of two values from attribute A_i or A_j, according to their corresponding comparison operators.

A *comparable dependency* (CD) φ with general comparable functions over a dataspace $\mathcal{S}$ is in the form of

$$\varphi : \quad \bigwedge \theta(A_i, A_j) \rightarrow \theta(B_1, B_2)$$

If two objects have comparable values on A_i or A_j, then they must have comparable values on B_1 or B_2.

Shaoxu Song sxsong@tsinghua.edu.cn

Example

Consider

$$\varphi_4 : \theta(\mathsf{manu}, \mathsf{prod}) \rightarrow \theta(\mathsf{tel}, \mathsf{phn})$$

where $\theta(\mathsf{tel}, \mathsf{phn})$ is $[\mathsf{tel} = \mathsf{tel}, \mathsf{tel} = \mathsf{phn}, \mathsf{phn} = \mathsf{phn}]$

- we have $(t_1, t_3) \asymp \mathrm{LHS}(\varphi_4)$ also agree $(t_1, t_3) \asymp \mathrm{RHS}(\varphi_4)$
- denoted by $(t_1, t_3) \models \varphi_4$.

t_1 : {(name : iPod), (color : red), (manu : Apple Inc.), (tel : 567), (addr : InfiniteLoop, CA), (website : itunes.com)};
t_2 : {(name : iPod), (color : cardinal), (prod : Apple), (tel : 123), (post : InfiniteLoop, Cupert), (website : apple.com)};
t_3 : {(name : iPad), (color : white), (manu : Apple Inc.), (post : InfiniteLoop), (website : apple.com), (phn : 567)}.

Shaoxu Song sxsong@tsinghua.edu.cn

Approximate Dependencies

Due to the extremely high heterogeneity, data dependencies might not exactly hold in a given dataspace.

$$\varphi_4 : \theta(\mathsf{manu}, \mathsf{prod}) \rightarrow \theta(\mathsf{tel}, \mathsf{phn}),$$

- e.g., $(t_1, t_2) \asymp \mathrm{LHS}(\varphi_4)$ but $(t_1, t_2) \not\asymp \mathrm{RHS}(\varphi_4)$
- i.e., $(t_1, t_2) \nvDash \varphi_4$

t_1 : {(name : iPod), (color : red), (manu : Apple Inc.), (tel : 567), (addr : InfiniteLoop, CA), (website : itunes.com)};
t_2 : {(name : iPod), (color : cardinal), (prod : Apple), (tel : 123), (post : InfiniteLoop, Cupert), (website : apple.com)};
t_3 : {(name : iPad), (color : white), (manu : Apple Inc.), (post : InfiniteLoop), (website : apple.com), (phn : 567)}.

Measure

To evaluate how a dependency "almost" holds in a data instance

- Error measure

$$g_3(\varphi, \mathcal{S}) = \frac{|\mathcal{S}| - \max\{|T| \mid T \subseteq \mathcal{S}, T \vDash \varphi\}}{|\mathcal{S}|},$$

the minimum number of objects that have to be removed from the dataspace $\mathcal{S}$ for a dependency φ to hold.

- Confidence measure

$$\mathsf{conf}(\varphi, \mathcal{S}) = \frac{\max\{|T| \mid T \subseteq \mathcal{S}, T \vDash \varphi\}}{|\mathcal{S}|}.$$

the maximum number of objects reserved after removing minimum objects of violations with respect to φ.

Shaoxu Song sxsong@tsinghua.edu.cn

Example

$$\varphi_4 : \theta(\mathsf{manu}, \mathsf{prod}) \rightarrow \theta(\mathsf{tel}, \mathsf{phn}),$$

Error measure

- $\{t_2\}$ is a minimum violation set w.r.t. φ_4
- such that all the remaining objects $\{t_1, t_3\}$ satisfy φ_4
- $g_3 = 1/3$

Confidence measure

- $\{t_1, t_3\}$ a maximum keeping set w.r.t. φ_4
- $\mathsf{conf} = 2/3$

t_1 : {(name : iPod), (color : red), (manu : Apple Inc.), (tel : 567), (addr : InfiniteLoop, CA), (website : itunes.com)};
t_2 : {(name : iPod), (color : cardinal), (prod : Apple), (tel : 123), (post : InfiniteLoop, Cupert), (website : apple.com)};
t_3 : {(name : iPad), (color : white), (manu : Apple Inc.), (post : InfiniteLoop), (website : apple.com), (phn : 567)}.

Outline

Shaoxu Song sxsong@tsinghua.edu.cn

Validation Problem

Unfortunately, computation of error or confidence is generally hard

Given

- a dataspace $\mathcal{S}$
- a dependency φ
- a measure requirement η

the validation problem is to decide whether or not the measure of φ over $\mathcal{S}$ satisfies η.

E.g., to determine whether $g_3(\varphi, \mathcal{S}) \leq 0.2$ or $\mathsf{conf}(\varphi, \mathcal{S}) \geq 0.8$.

Theorem

The error and confidence validation problems are NP*-complete.*

The Hardness

The transitivity cannot be assumed, i.e., from $(t_1, t_2) \asymp \theta(A_i, A_i)$ and $(t_2, t_3) \asymp \theta(A_i, A_i)$ it does not necessarily follow that $(t_1, t_3) \asymp \theta(A_i, A_i)$.

$$t_1 : \{(A_1 : abc), \ldots\};$$
$$t_2 : \{(A_1 : abcd), \ldots\};$$
$$t_3 : \{(A_1 : abcde), \ldots\}.$$

E.g., $\theta(A_1, A_1) : [A_1 \approx_{\leq 1} A_1]$ with edit distance as metric d

- $d(t_1[A_1], t_2[A_1]) = 1 \leq 1$
- $d(t_2[A_1], t_3[A_1]) = 1 \leq 1$,
- but $d(t_1[A_1], t_3[A_1]) = 2 > 1$, that is, $(t_1, t_3) \not\asymp \theta(A_1, A_1)$.

The efficient validation computation based on disjoint grouping cannot be applied in this case of comparable functions.

Shaoxu Song sxsong@tsinghua.edu.cn

Approximation Computation

Compute an approximate error/confidence measure of φ over $\mathcal{S}$

- the approximate measure has a relative performance guarantee compared with exact measure,
- e.g., $\hat{g}/g \leq \rho$
- where $\hat{g}$ is an approximation of exact error measure g and ρ is approximation ratio

Shaoxu Song sxsong@tsinghua.edu.cn

Greedy Algorithm

- greedily count both objects when a violation occurs
- the complexity is $O(n^2)$

The error approximation

- outputs an estimate $\hat{g}$ with a bound $g \leq \hat{g} \leq 2g$ compared with the exact error measure g

Theorem

The confidence has no constant-factor approximation unless P=NP

- confidence is NP-hard to approximate within a constant factor
- g_3 error and confidence are not equivalent in an approximation-preserving way

Shaoxu Song sxsong@tsinghua.edu.cn

Randomized Algorithm

Greedy algorithm still has to consider all the objects in a dataspace.

Randomized algorithm evaluates just a small subset of objects

- randomly draw m samples
- estimate the error/confidence measure by using the violations to these m samples
- the estimate measure is still guaranteed by certain approximation bound with high probability
- e.g., $\Pr[\hat{g} \leq \rho g + \epsilon] \geq \delta$
- ϵ is an additive error
- δ is a probability guarantee
- m is determined by ϵ and δ

Outline

Shaoxu Song sxsong@tsinghua.edu.cn

Discovery Problem

The strict dependency discovery problem

- find a canonical cover of all dependencies that hold in data
- a canonical cover can be exponential in # of attributes
- high dimensionality in dataspaces (attributes and comparable functions)
- highly non-trivial (if not impossible) to discover a canonical cover of all dependencies

The k-length dependencies

- contain k or less comparable functions
- motivated by the concept of mining k-length itemsets in association rules

Shaoxu Song sxsong@tsinghua.edu.cn

Pay-as-you-go Discovery

In previous work of dataspaces, comparable attributes are identified in a *pay-as-you-go* style.

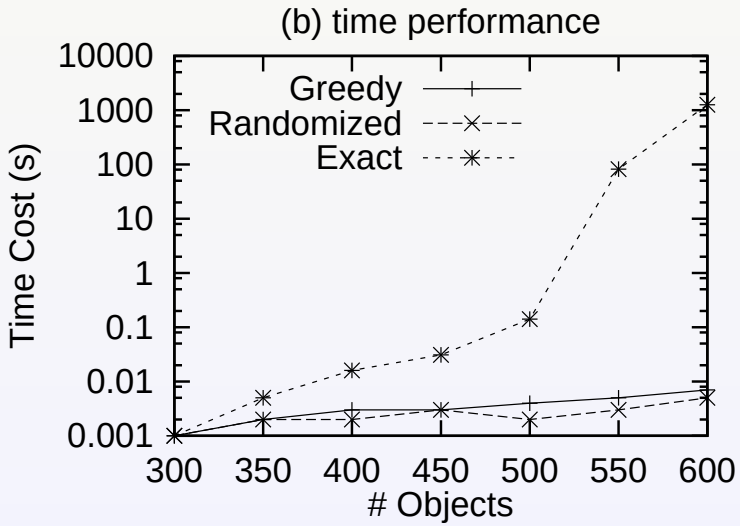
The discovery of dependencies should be conducted in an incremental way as well.

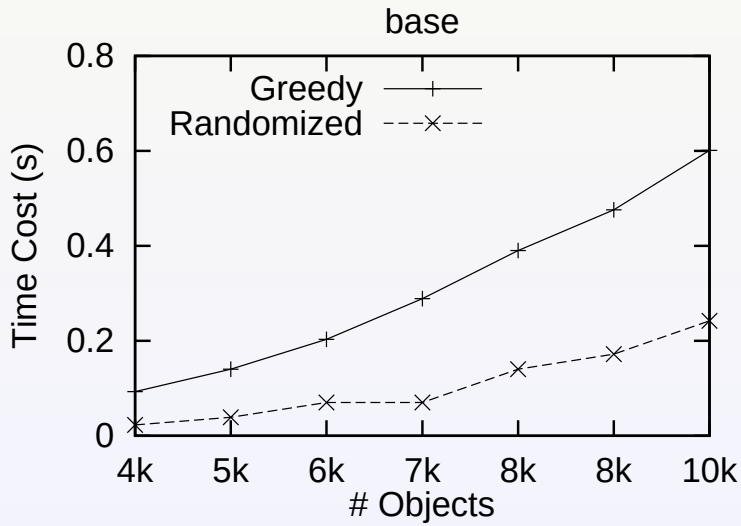
- given a set Σ of currently discovered dependencies
- and a newly identified comparable functions $\theta(A_i, A_j)$,
- we can generate new dependencies w.r.t. $\theta(A_i, A_j)$ according to the augmentation property

Validation Evaluation

Experiments on real data sets

- compute the exact/approximate measures of dependencies
- observe the corresponding performance
- exact computation does not scale well
- approximation computation keeps significantly lower time cost and scale well



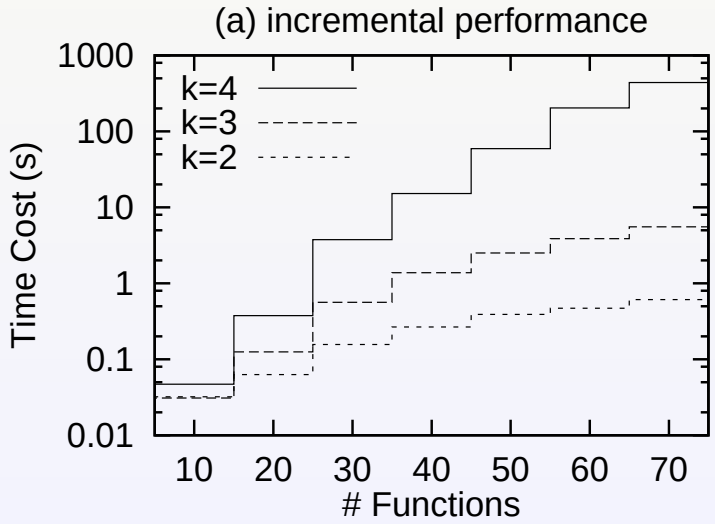


Shaoxu Song sxsong@tsinghua.edu.cn

Discovery Evaluation

Illustrate the incremental discovery of dependencies with the increase of functions.

- y-axis is scaled logarithmically
- time cost increases heavily with the number of functions
- the intrinsic hardness in discovering dependencies with respect to attributes (and the corresponding comparable functions)
- size k of functions also affects the discovery performance largely

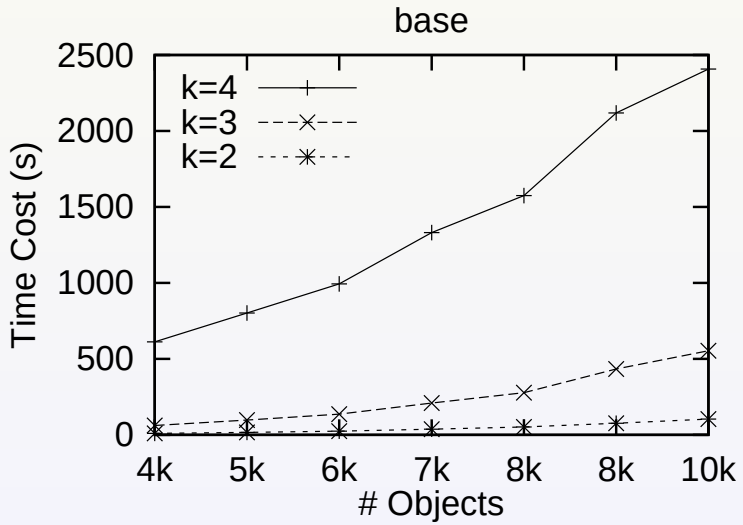


Shaoxu Song sxsong@tsinghua.edu.cn

Discovery Evaluation

The discovery algorithm scales well in large size of objects

- greedy approximation is adopted for validation
- verify the efficiency of approximation computation proposed for validating dependencies



Shaoxu Song sxsong@tsinghua.edu.cn

Conclusion

This is the first work to adapt dependencies to dataspaces with the consideration of comparable attribute values

- it is already hard to tell whether a dependency almost holds in the data
- the confidence validation is also proved hard to approximate to within any constant factor
- propose several greedy and randomized approaches for approximately solving the validation problem
- study the pay-as-you-go discovery of dependencies from dataspaces