HAL
archives-ouvertes.fr

Discovering Injective Mapping Between Relations in Astrophysics Databases

Nicu Stancioiu, Lhouari Nourine, Jean-Marc Petit, Vasile-Marian Scuturici, Dominique Fouchez, Emmanuel Gangler, Philippe Gris

▶ **To cite this version:**

Nicu Stancioiu, Lhouari Nourine, Jean-Marc Petit, Vasile-Marian Scuturici, Dominique Fouchez, et al.. Discovering Injective Mapping Between Relations in Astrophysics Databases. Information Search, Integration, and Personalization 11th International Workshop, Nov 2016, Lyon, France. hal-01672571

HAL Id: hal-01672571
https://hal.archives-ouvertes.fr/hal-01672571

Submitted on 26 Dec 2017

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Discovering Injective Mapping between Relations in Astrophysics Databases

Nicu-Razvan Stancioiu[1], Lhouari Nourine[2],
Jean-Marc Petit[1], Vasile-Marian Scuturici[1],
Dominique Fouchez[4], Emmanuel Gangler[3], and Philippe Gris[3]

[1]Université de Lyon, CNRS
INSA-Lyon, LIRIS, UMR5205, F-69621 Villeurbanne, France
E-mail: {nicu.stancioiu,marian.scuturici,jean-marc.petit}@insa-lyon.fr

[2]Clermont-Université, Université Blaise Pascal, LIMOS, CNRS, France
E-mail: nourine@isima.fr

[3]Clermont-Université, LPC, CNRS, France
E-mail: emmanuel.gangler@clermont.in2p3.fr

[4]IN2P3, CNRS, Marseile, France
E-mail: fouchez@cppm.in2p3.fr

Abstract. Data in Astrophysics are very often structured with the relational data model. One particularity is that every value is a real number and comes with an associated error measure, leading to a numerical interval $[value - error, value + error]$. Such Astrophysics databases can be seen as *interval-based numerical databases*.
Classical data mining approach, specifically those related to integrity constraints, are likely to produce useless results on such databases, as the strict equality is very unlikely to give meaningful results.
In this paper, we revisit a well-known problem, based on unary inclusion dependency discovery, to match the particularities of Astrophysics Databases. We propose to discover injective mapping between attributes of a source relation and a target relation. At first, we define two notions of inclusion between intervals. Then, we adapt a condensed representation proposed in [15] allowing to find a mapping function between the source and the target. The proposition has been implemented and several experiments have been conducted on both real-life and synthetic databases.

1 Introduction

Astrophysics is known to generate huge amount of data in large experiments, as for example with the Large Synoptic Survey Telescope (LSST[1]), a wide-field survey reflecting telescope in Chile. The camera is expected to take over 500,000 pictures per year, leading to more than 60 petabytes of data at the end of

[1] https://www.lsst.org

d	u	g	r	i	z	err_u	err_g	err_r	err_i	err_z
t_1 :	23.37	24.52	23.42	24.27	20.76	0.52	0.52	0.43	0.73	0.21
t_2 :	18.44	16.33	15.48	15.20	15.07	0.01	0.01	0.01	0.02	0.01
t_3 :	23.77	22.85	23.39	22.41	22.60	0.77	0.16	0.41	0.29	0.76
t_4 :	22.72	20.88	19.51	18.79	18.41	0.43	0.03	0.01	0.01	0.03
t_5 :	22.48	21.82	21.38	21.17	21.25	0.24	0.07	0.06	0.07	0.29
t_6 :	24.04	20.99	19.68	19.10	18.84	0.77	0.03	0.01	0.01	0.03

Fig. 1: Example of astrophysics database

the project. After a long image processing process, relevant data are stored in specialized Relational Database Management Systems (RDBMS). Since every value comes with its associated error measure, it can be seen as a numerical interval $[value - error, value + error]$. Then, such Astrophysics databases are *interval-based numerical databases* and look like the data given in Figure 1. u and err_u (resp. g, r, i, z) are the magnitude (flux in log scale) and corresponding error of an astrophysical object measured through a passband color filter named u (resp. g, r, i, z). The u, g, r, i, z color filters slice the visible spectrum in five similar size bins. For each attribute that contains a magnitude parameter, there is an associated error measure attribute (e.g. u and err_u).

Classical data mining approach, specifically those related to integrity constraints like functional dependencies (FD), conditional FD or inclusion dependencies (IND), are likely to produce useless results on such databases. The particularities of interval-based databases impose peculiar problems to the associated discovery problems.

In this paper, we revisit a well-known problem, the discovery of unary inclusion dependency discovery, to match the particularities of Astrophysics Databases. Due to the nature of the data in use, the proposed solution has to be changed with respect to existing approaches [15,16].
More precisely, we propose to discover injective mapping between attributes of a source relation and a target relation. In order to solve the problem of discovering such mappings, we adapt the problem of discovering unary inclusion dependencies for interval-based databases. With respect to the application domain in Astrophysics, we do not claim that such mappings solve real problems for astrophysicists, even if some applications could benefit from it. The problem studied in this paper should be thought as a first step, opening many opportunities to address others related problems, more challenging and interesting for astrophysicists.

Let us consider two interval-based relations s and t over relation schema S and T respectively. As usual, attributes of a relation r over R are denoted by $sch(R)$.

Problem statement

> Given a source s and a target t such that $|sch(S)| \leq |sch(T)|$, find a mapping f from $sch(S)$ to $sch(T)$ such that (1) f is injective ($f(A) =$

$f(B) \Rightarrow A = B$) and (2) for every attribute $A \in sch(R)$, the values of A in s and the values of $f(A)$ in t should be as similar as possible, i.e. some forms of inclusion dependencies should exist between them.

To deal with this problem, we propose a contribution based on the following three-step process:

- First, we define two types of membership of an interval into a collection of intervals. The first one is based on the classical inclusion between intervals while the second one is defined on the so-called *canonical representation* of a collection of intervals.
- Second, we extend the work of [15] and build a condensed representation of an interval-based numerical database as a binary relation (or transactional database where transactions are values and items are attributes). At the end, we discover a set of approximate unary inclusion dependencies from the source to the target.
- Finally, we propose a method to find an injective mapping, which is equivalent to the minimum weight matching in an weighted bipartite graph, resolved with the *Hungarian* algorithm [12] in this paper.

The proposition has been implemented and several experiments have been conducted on both real-life and synthetic databases. Even if the overall complexity of the studied problem remains polynomial, the overhead with respect to classical databases turns out to be rather low. The main lesson we have learned from this work is that many contributions in pattern mining could be revisited in order to deal with interval-based numerical databases.

To the best of our knowledge, this is the first contribution dealing with the discovery of integrity constraints in interval-based numerical databases.

Paper Organization . The remaining part of the paper is organized as follows: Section 2 gives preliminaries of the paper. Section 3 adapts the condensed representation of [15] to interval-based databases. Section 4 introduces details about the discovery of approximate unary inclusion dependencies. Section 5 describes the main algorithm *SR2TR* providing an injective mapping between a source and a target relations, and the results of experiments. Section 7 concludes the paper and gives some perspectives to this work.

2 Preliminaries

Basic database notions are given here, more details can be found for example in [14]. We restrict our attention to interval-based numerical databases only.

Let $\mathcal{U}$ be a set of attributes and $\mathcal{D}$ the possible intervals over real numbers. A relation symbol is generally denoted by R and its schema by $sch(R), sch(R) \subseteq \mathcal{U}$. When clear from context, we shall use R instead of $sch(R)$. Each attribute has a domain, included in $\mathcal{D}$. A tuple over R is an element of the cartesian product $\mathcal{D}^{|R|}$. An interval-based numerical relation (or simply relation) r over R is a set

of tuples over R. An interval-based numerical database d (or simply database) over a set of relation symbol $\{R_1, \ldots, R_n\}$ is a set of n interval-based relations $\{r_1, \ldots, r_n\}$, r_i defined over R_i for $i = 1..n$.

Given a relation r over R and $A \in R$, the active domain of A in r is denoted by $ADOM(A, r)$. The active domain of r is denoted by $ADOM(r) = \bigcup_{A \in R} ADOM(A, r)$. The projection of a tuple t on an attribute set $X \subseteq R$ is denoted by $t[X]$. The projection of a relation r onto a set of attributes X, denoted by $\pi_X(r)$, is defined by $\pi_X(r) = \{t[X] | t \in r\}$.

Let I, J be two intervals of $\mathcal{D}$. We note $min(I)$ (resp. $max(I)$) the minimum (resp. maximum) value of I. I is contained in J, denoted by $I \subseteq J$, if $min(J) \leq min(I) \leq max(I) \leq max(J)$.

Let $\mathcal{I}$ be a collection of intervals of $\mathcal{D}$. The union of $\mathcal{I}$ is the minimal number of intervals covering $\mathcal{I}$. Since its union is unique, it represents a canonical form of $\mathcal{I}$, and will be denoted by $cano(\mathcal{I})$. More formally, $cano(\mathcal{I})$ can be defined by induction as follows:

$$\begin{aligned} cano(\mathcal{I}) &= \mathcal{I} && \text{if for all } J_1, J_2 \in \mathcal{I}, J_1 \cap J_2 = \emptyset \\ &= cano(\mathcal{I} \setminus \{J_1, J_2\}) \cup \{J_3\}) && \text{otherwise with } J_1, J_2 \in \mathcal{I}, J_1 \cap J_2 \neq \emptyset \text{ and} \\ & && J_3 = [min(min(J_1), min(J_2)), max(max(J_1), max(J_2))] \end{aligned}$$

$\mathcal{I}$ is said to be *connected* if $|cano(\mathcal{I})| = 1$.

We now introduce the classical syntax and semantics of unary inclusion dependencies between relation symbols R and S.

Definition 1. *An* unary inclusion dependency *(UIND) from R to S is a statement of the form $R[A] \subseteq S[B]$, where $A \in R$, $B \in S$.*

Definition 2. *Let $d = \{r, s\}$ be a database over $\{R, S\}$. An unary inclusion dependency $R[A] \subseteq S[B]$ is satisfied in d, denoted by $d \models R[A] \subseteq S[B]$, iff for all $u \in r$, there is $v \in s$ such that $u[A] = v[B]$ or equivalently $\pi_A(r) \subseteq \pi_B(s)$.*

Example 1. Let r_0 be a classical relation over R (see Figure 2).

r_0	A	B	C	D
t_1	0	1	1	2
t_2	1	2	2	1
t_3	2	1	1	0

Fig. 2: A toy relation r_0

Several classical UINDs are satisfied in $\{r_0\}$, for instance $R[A] \subseteq R[D]$ and $R[B] \subseteq R[C]$.

When working with intervals, the strict equality "=" used in the definition of the satisfaction of an UIND is likely to produce unsatisfying results. For instance, in Figure 3, the relation r_1 has no satisfied UINDs.

r_1	A	B	C
t_1 :	[0,0.5]	[0,1]	[0,1.5]
t_2 :	[1,1.5]	[1,2]	[3,3.5]
t_3 :	[2,3]	[1.5,4]	[2,3]

Fig. 3: A toy interval-based relation r_1

This leads to introducing different UIND satisfaction on intervals.
We define two kinds of satisfied UINDs over interval-based numerical database: *classical satisfaction* based on interval inclusion over collection of intervals and *canonical satisfaction* based on interval inclusion over the canonical representation of collection of intervals.
Let $d = \{r, s\}$ be an interval-based numerical database over $\{R, S\}$.

Definition 3. *An UIND $R[A] \subseteq S[B]$ is classically satisfied in d, denoted by $d \models_1 R[A] \subseteq S[B]$, iff for all $u \in r$, there is $v \in s$ such that $u[A] \subseteq v[B]$.*
An UIND $R[A] \subseteq S[B]$ is canonically satisfied in d, denoted by $d \models_2 R[A] \subseteq S[B]$, iff for all $u \in r$, there is $v \in cano(ADOM(B, s))$ such that $u[A] \subseteq v$.

We will note $d \models_\lambda R[A] \subseteq S[B]$ to refer to both of them.

Example 2. In the relation r_1 of figure 3, for the satisfaction, we have an UIND $r_0 \models A \subseteq_1 B$ as $[0, 0.5] \subseteq [0, 1]$, $[1, 1.5] \subseteq [1, 2]$ and $[2, 3] \subseteq [1.5, 4]$. For the second one, $r_0 \models A \subseteq_2 B$ as $[0, 0.5] \subseteq [0, 4]$, $[1, 1.5] \subseteq [0, 4]$ and $[2, 3] \subseteq [0, 4]$.

We also need to define when a given interval belongs to a collection of intervals.
Let I be an interval and $\mathcal{I}$ a collection of intervals.
I is classically *included* in $\mathcal{I}$, denoted by $I \subseteq_1 \mathcal{I}$, if there exists $I' \in \mathcal{I}$ such that $I \subseteq I'$. I is canonically *included* in $\mathcal{I}$, denoted by $I \subseteq_2 \mathcal{I}$ if $I \subseteq_1 cano(\mathcal{I})$.

3 Condensed representation for interval-based relations

We now extend the contribution for discovering UINDs in databases [15] to interval-based numerical databases. This is, up to our knowledge, the best approach for discovering UINDs. It relies on a preprocessing to get a condensed representation from the initial database.
Now, we define a condensed representation for UIND discovery.

Definition 4. *The condensed representation of an interval-based numerical relation r, denoted by $CR_{\subseteq_\lambda}(r)$, is defined by:*

$$CR_{\subseteq_\lambda}(r) = \{(I, X) \mid I \in ADOM(r), X = \{A \in R \mid I \subseteq_\lambda ADOM(A, r)\}\}$$

Condensed representations from the two defined satisfactions can be different. We denote $CR_{\subseteq_1}(r)$ and $CR_{\subseteq_2}(r)$ the condensed representations for $\subseteq_1$ and $\subseteq_2$ respectively.

Example 3. The condensed representations $CR_{\subseteq_1}(r_1)$ and $CR_{\subseteq_2}(r_1)$ of the relation r_1 (see Figure 3).

$CR_{\subseteq_1}(r_1)$	
[0,0.5]	ABC
[1,1.5]	ABC
[2,3]	ABC
[0,1]	BC
[1,2]	B
[1.5,4]	B
[0,1.5]	C
[3,3.5]	BC

$CR_{\subseteq_2}(r_1)$	
[0,0.5]	ABC
[1,1.5]	ABC
[2,3]	ABC
[0,1]	BC
[1,2]	B
[1.5,4]	B
[0,1.5]	BC
[3,3.5]	BC

Fig. 4: Condensed representations of r_1 for both semantics

Given a set of relations $r_1, r_2, ..., r_n$, its condensed representation is defined by as:

$$CR_{\subseteq_\lambda}(r_1, r_2, ..., r_n) = \bigcup_{i=1..n} CR_{\subseteq_\lambda}(r_i).$$

Definition 5. *The support of an attribute set $X \subseteq R$ in $CR_{\subseteq_\lambda}(r)$, denoted by $sup(X, CR_{\subseteq_\lambda}(r))$, is defined by:*

$$sup(X, CR_{\subseteq_\lambda}(r)) = |\{(i, Y) \in CR_{\subseteq_\lambda}(r) | X \subseteq Y\}|$$

Definition 6. *The closure of an attribute $A \in sch(R)$ with respect to $CR_{\subseteq_\lambda}(r)$, denoted by $A^+_{CR_{\subseteq_\lambda}(r)}$, is defined as:*

$$A^+_{CR_{\subseteq_\lambda}(r)} = \bigcap_{(i,X) \in CR_{\subseteq_\lambda}(r)} \{X | A \in X\}$$

Example 4. In *Figure 4*, for $CR_{\subseteq_1}(r_1)$ we have that $sup(\{A\}, CR_{\subseteq_1}(r_1)) = 3$, $sup(\{A, B\}, CR_{\subseteq_1}(r_1)) = 3$ and $A^+_{CR_{\subseteq_1}(r_1)} = \{A, B, C\}$, $C^+_{CR_{\subseteq_1}(r_1)} = \{C\}$. As for $CR_{\subseteq_2}(r_1)$, $sup(\{B, C\}, CR_{\subseteq_2}(r_1)) = 6$, $sup(\{C\}, CR_{\subseteq_2}(r_1)) = 6$ and $C^+_{CR_{\subseteq_2}(r_1)} = \{B, C\}$.

4 Unary Inclusion Dependencies Discovery in a single interval-based relation

To alleviate the notations, we consider a single relation only, i.e. UIND of the form $r \models_\lambda R[A] \subseteq R[B]$. Let r be a relation over R and $A, B \in R$.
We first give a technical lemma which relates the definition of $r \models_2 A \subseteq B$ to the canonical representation of the intervals of A in r.

Lemma 1. $r \models_2 A \subseteq B \Longleftrightarrow \forall I \in cano(ADOM(A,r)), \exists J \in cano(ADOM(B,r)), I \subseteq J$

Proof. ($\Longrightarrow$) Suppose not. Assume, to the contrary that there exists $I' \in cano(ADOM(A,r))$, such that for all $J \in cano(ADOM(B,r))$, $I' \not\subseteq J$.
Let $\mathcal{I} \subseteq ADOM(A,r)$ be the maximal collection of intervals such that $cano(\mathcal{I}) = \{\mathcal{I}'\}$.
By definition, $r \models_2 A \subseteq B$ implies that for all $I \in ADOM(A,r)$, there exists $J \in cano(ADOM(B,r))$ such that $I \subseteq J$.
If $|\mathcal{I}| = 1$, then we have a contradiction and the result follows.
Assume $|\mathcal{I}| > 1$. The collection $\mathcal{I}$ can be divided into $n > 1$ disjoint non-empty collections of intervals $\{\mathcal{I}_1, \mathcal{I}_2,..,\mathcal{I}_n\}$ such that there exist n different associated intervals $\{J_1, J_2, .., J_n\} \in cano(ADOM(B,r))$ such that for all $I \in \mathcal{I}_\lambda$, $I \subseteq J_\lambda$, $\lambda \in \{1, 2, .., n\}$.
Since $|cano(\mathcal{I})| = 1$, for every $\mathcal{I}_\lambda \in \{\mathcal{I}_1, \mathcal{I}_2,..,\mathcal{I}_n\}$, there exists $\mathcal{I}_{\lambda'} \in \{\mathcal{I}_1, \mathcal{I}_2,..,\mathcal{I}_n\}, \lambda \neq \lambda'$ such that there exists $I_0 \in \mathcal{I}_\lambda$, $K_0 \in \mathcal{I}_{\lambda'}$ such that $I_0 \cap K_0 \neq \emptyset$. But $I_0 \subseteq J_\lambda$, $K_0 \subseteq J_{\lambda'}$, then $J_\lambda \cap J_{\lambda'} \neq \emptyset$. Contradiction as $J_\lambda, J_{\lambda'}$ are supposed to be non-intersecting.
($\Longleftarrow$)
Let $I' \in ADOM(A,r)$. Then there exists $I \in cano(ADOM(A,r))$ such that $I' \subseteq I$. Since for all $I \in cano(ADOM(A,r))$, there exists $J \in cano(ADOM(B,r))$ such that $I \subseteq J$, it follows that $I' \subseteq J$. Thus $r \models_2 A \subseteq B$.

Intuitively, the main result of the paper states that every inclusion of the form $A \subseteq B$ that holds in r turns out to be equivalent to a closure computation on the associated condensed representation.
We can now give the main result of the paper.

Theorem 1.

$$r \models_\lambda A \subseteq B \Longleftrightarrow B \in A^+_{CR_{\subseteq_\lambda}(r)}$$

Proof. We consider the two UIND satisfactions presented before:

1. $r \models_1 A \subseteq B \Longleftrightarrow B \in A^+_{CR_{\subseteq_1}(r)}$
   ($\Longrightarrow$)
   Let $(I', X) \in CR_{\subseteq_1}(r)$ such that $A \in X$. Then, there exists $I \in ADOM(A,r)$ such that $I' \subseteq I$. Or $r \models_1 A \subseteq B$ implies that there exists $J \in ADOM(B,r)$ such that $I \subseteq J$. It follows that $I' \subseteq J$, and thus $B \in X$. Then $B \in A^+_{CR_{\subseteq_1}(r)}$.

($\Longleftarrow$)
$B \in A^+_{CR_{\subseteq_1}(r)} \Longleftrightarrow$ for all $(I, X) \in CR_{\subseteq_1}(r)$, if $A \in X$ then $B \in X$. For all $I \in ADOM(A, r)$, there exists a pair $(I, X) \in CR_{\subseteq_1}(r)$ such that $A \in X$. Thus, $B \in X$ also holds, i.e. there exists $J \in ADOM(B, r)$ such that $I \subseteq J$. Thus $r \models_1 A \subseteq B$.

2. $r \models_2 A \subseteq B \Longleftrightarrow B \in A^+_{CR_{\subseteq_2}(r)}$
($\Longrightarrow$)
Let $(I', X) \in CR_{\subseteq_2}(r)$ such that $A \in X$. Then, there exists $I \in cano(ADOM(A, r))$ such that $I' \subseteq I$. Based on lemma 1, $r \models_2 A \subseteq B$ implies that there exists $J \in cano(ADOM(B, r))$ such that $I \subseteq J$. It follows that $I' \subseteq J$, and thus $B \in X$. Then $B \in A^+_{CR_{\subseteq_1}(r)}$.
($\Longleftarrow$)
For all $I' \in ADOM(A, r)$, there exists a pair $(I, X) \in CR_{\subseteq_2}(r)$ such that $A \in X$ and $I' \subseteq I$. Thus, $B \in X$ also holds, i.e. there exists $J \in ADOM(B, r)$ such that $I \subseteq J$. It follows that $I' \subseteq J$.
Thus $r \models_2 A \subseteq B$.

From previous theorem, the discovery of UIND is based on the following property, based on support counting in the condensed representation.

Property 1.

$$B \in A^+_{CR_{\subseteq_\lambda}(r)} \Longleftrightarrow sup(\{A, B\}, CR_{\subseteq_\lambda}(r)) = sup(\{A\}, CR_{\subseteq_\lambda}(r)).$$

Proof. $B \in A^+_{CR_{\subseteq_\lambda}(r)} \Longleftrightarrow$ for all $(I, X) \in CR_{\subseteq_\lambda}(r)$, if $A \in X$ then $B \in X$. Equivalently, the support of $\{A\}$ and $\{A, B\}$ are the same in $CR_{\subseteq_\lambda}(r)$, i.e. $sup(\{A, B\}, CR_{\subseteq_\lambda}(r)) = sup(\{A\}, CR_{\subseteq_\lambda}(r))$.

5 Approximate unary IND between two interval-based relations

Within this setting, an unary IND $A \subseteq_\lambda B$ may be still unsatisfied in a relation due to a few counter-examples. As a result, we introduce an approximation measure to extract approximate unary inclusion dependencies from a relation. This approximation will be calculated based on the support of attributes' sets. The error measure related to the correspondence between two attributes, denoted *error* can be defined using the support of attribute sets as follows:

Definition 7. *Let* $\boldsymbol{r}$ *be a relation over* R *and* $A, B \in R$.

$$error(r \models_\lambda A \subseteq B) = 1 - \frac{sup(\{A, B\}, CR_{\subseteq_\lambda}(r))}{sup(\{A\}, CR_{\subseteq_\lambda}(r))}$$

Given two relations r over R and s over S, we are interested in finding the approximative unary inclusion dependencies from single attributes of R with

respect to singles attributes of S. From now on, we consider the relation r as a *source* relation and s as a *target* relation.
Based on the previous definition we can build a matrix of error measures, such that each element of the matrix is represented as the value of the error measure between a single attribute from R and a single attribute of S.

Example 5. Let r_2, s_2 be two toy relations:

r_2	A	B	C
t_1 :	[0,0.5]	[0,1]	[0,1]
t_2 :	[1,1.5]	[1,2]	[3,3.5]
t_3 :	[2,3]	[1.5,4]	[2,3]

s_2	D	E	F	G
t_1 :	[0,1.5]	[1,2]	[1.6,1.8]	[2,3]
t_2 :	[0,1.5]	[1,2]	[3.1,4]	[2,3]
t_3 :	[0,1.5]	[0,1]	[0,0.5]	[1,2]
t_4 :	[2,3.5]	[2,3]	[0,0.5]	[1,2]
t_5 :	[2,3.5]	[1,2]	[0,0.5]	[1,2]

Fig. 5: Toy relations r_2 and s_2

Based on the condensed representation $CR_{\subseteq_1}(r_2, s_2)$, we build a matrix of error measures $error(\{r_2, s_2\} \models_1 X \subseteq Y)$ with X as attribute over r_2 and Y as attribute over s_2:

$CR_{\subseteq_1}(r_2, s_2)$	
[0,0.5]	ABCDEF
[1,1.5]	ABDEG
[2,3]	ABCDEG
[0,1]	BCDE
[1,2]	BEG
[1.5,4]	B
[3,3.5]	BCD
[0,1.5]	D
[2,3.5]	BD
[1.6,1.8]	BEFG
[3.1,4]	BF

$error(\{r_2, s_2\} \models_1 X \subseteq Y)$	D	E	F	G
A	0.000	0.000	0.667	0.333
B	0.400	0.400	0.700	0.600
C	0.000	0.250	0.750	0.750

Fig. 6: The condensed representation of r_2 and s_2 and the associated matrix of error measures

As we search to find the most appropriate injective matching function between two schema relations, given the matrix of error measures, we can reformulate this problem as follows:

> Given an error matrix between two schema relations R and S, find the best matching with the minimum error between R and S.

This problem is in fact an *assignment problem* [12] which consists of finding a maximum weight matching in a weighted bipartite graph. *Hungarian Algorithm* [11] is one of the algorithms that can solve the *assignment problem*. There are other algorithms that include adaptations as the *Simplex Algorithm* and the *Auction Algorithm* [2]. The assignment problem is a special case of the *transportation problem* [10], which is a variation of *minimum cost maximum flow problem* [1].

6 Scalable algorithms

We propose a polynomial algorithm called $SR2TR$, which provides an injective mapping based on the discovery of a set of approximate unary INDs from a source relation r over R to a target relation s over S.

Algorithm 1 (SR2TR) Mapping from Source Relation to Target Relation

Input: two relations: r over R and s over S
Output: A mapping function f from R to S
1: CR = Preprocessing(r, s)
2: M = MEM(CR)
3: f = FindMatching(M)
4: **return** f.

Algorithm 1 can be summarized as follows:

1. Process the condensed representation $CR(r, s)$ - *Preprocessing(r,s)*;
2. Build the matrix of error measures M based on $CR(r, s)$ - *MEM(CR)*;
3. Find the minimum weight matching in the weighted bipartite graph constructed from M - *FindMatching(M)*.

Two other algorithms are provided: Algorithm 2 for the *Preprocessing* function (line 1 of Algorithm 1) and Algorithm 3 for the MEM function (line 2 of Algorithm 1). The $FindMatching$ function (line 3, Algorithm 3) corresponds to the *minimum weight matching in a weighted bipartite graph*. From the matrix of error measures M, we have implemented the *Hungarian Algorithm*, not detailed here.

Algorithm 2 (Preprocessing) Computing the condensed representation of $r \cup s$

Input: Two relations r over R and s over S
Output: Condensed representation $CR(r, s)$
1: CR=$\emptyset$
2: **for all** $I \in ADOM(r) \cup ADOM(s)$ **do**
3: $BR = \emptyset$
4: **for all** $A \in R \cup S$ **do**
5: **if** $check\_inclusion(I, ADOM(A, r \cup s)) = true$ **then**
6: $BR[A] = true$
7: **else**
8: $BR[A] = false$
9: **end if**
10: **end for**
11: $CR.add(BR)$
12: **end for**
13: **return** CR

For each interval $I \in ADOM(r) \cup ADOM(s)$, we search all single attributes $A \in R \cup S$ for which I is *included* in $ADOM(A, r \cup s)$ (line 5). Then we update the binary relation BR accordingly, which is afterwards added to CR. As the set $ADOM(A, r \cup s)$ is a set of intervals, from which we can construct an *interval tree* for each single attribute in $R \cup S$, so that $check\_inclusion$ is logarithmic in the size of the r and s.

Algorithm 3 (MEM) Compute the matrix of error measures

Input: Condensed representation $CR(r, s)$
Output: The matrix of error measures between R and S
1: $init(M)$
2: **for all** $A \in R$ **do**
3: **for all** $(I, X) \in CR(r, s)$ where $A \in X$ **do**
4: $sup1[A] = sup1[A] + 1$
5: **for all** $B \in S$ where $B \in X$ **do**
6: $sup2[A][B] = sup2[A][B] + 1$
7: **end for**
8: **end for**
9: **end for**
10: **for all** $A \in R$ and $B \in S$ **do**
11: $M[A][B] = 1.0 - sup2[A][B]/sup1[A]$
12: **end for**
13: **return** M

Based on the condensed representation $CR(r, s)$, we form the matrix of error measures M of size $|R| * |S|$. Line 1, M is initialized and filled with zeros in the $init(M)$ function. At this point we create two arrays, $sup1$ and $sup2$, referring

to the support of size 1 and 2 of $R \cup S$. Then, we search through all the elements of CR and we update step by step the support accordingly (lines 4, 6). Lines 10-11, we fill the matrix M, such that the value of an element $M[A][B]$ of the matrix is equal to $1 - \frac{sup(\{A,B\})}{sup(\{A\})}$, where $A \in R$ and $B \in S$.

Complexity analysis of SR2TR Let $n = |R|$, $m = |S|$, $a = |r|, b = |s|$. The theoretical complexity is in $O((n * a + m * b) * (n * log(a) + m * log(b)) + P)$ where P is the complexity of the matching algorithm. In case of the Hungarian Algorithm, its complexity would be $(n+m)^4$, which can be reduced to $(n+m)^3$ [4,8,9,13].

7 Experimental Results

We implemented the previous algorithms in C++ and conducted experiments to determine its effectiveness. We used datasets provided by astrophysicists of IN2P3 and synthetic databases to check the scalability of the algorithm. Our experiments were run using a machine with an Intel Core i7-4712MQ (2.3 GHz) CPU and 12GB of memory. We focused on the classical UIND definition, referred to as $\subseteq_1$. The results with canonical UIND, referred to as $\subseteq_2$, being quite similar are not discussed.
The IN2P3 dataset is composed of two databases with 11 single attributes each.

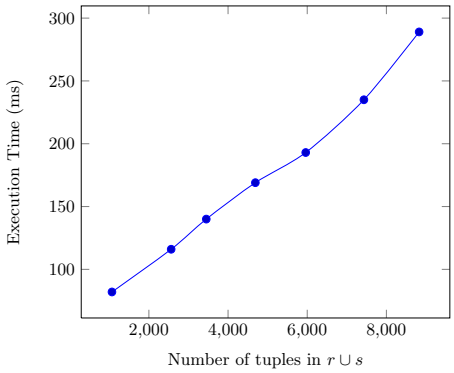

Fig. 7: Real-life astrophysics database

With respect to the number of tuples of the two databases, we obtained acceptable execution times (see Figure 7), linear in the size of the data.
To test the scalability of our technique with regard to the response time and the memory usage, we created 4 tests realized on synthetic datasets. The datasets are composed of floating numbers in the interval [0,5000] and with an error measure in the interval [0,1].

The first test considers a target relation of 100000 tuples, 20 attributes in each relation schema and a varying number of tuples in the source relation (see Figure 8). We can observe a polynomial behavior on experimental results both in response time and memory usage.

The second test considers a source relation of 1000 tuples, 20 attributes in each relation schema and a varying number of tuples in the target relation (see Figure 9). In both the response time and memory usage we recognize a linear behavior.

The following test considers a target schema relation with 100 single attributes, 100 tuples in each relation and a varying number of single attributes in the source schema relation (see Figure 10). We observe a linear behavior on experimental results both in response time and memory usage.

The last test considers a source relation with 10 single attributes, and 100 tuples in each relation and a varying number of single attributes in the target schema relation (see Figure 11). A polynomial behavior can be observed both in response time and memory usage.

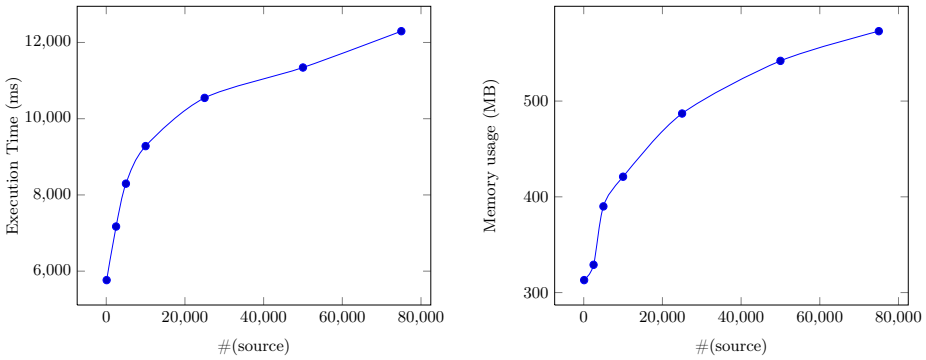

Fig. 8: Varying the number of tuples in the source relation

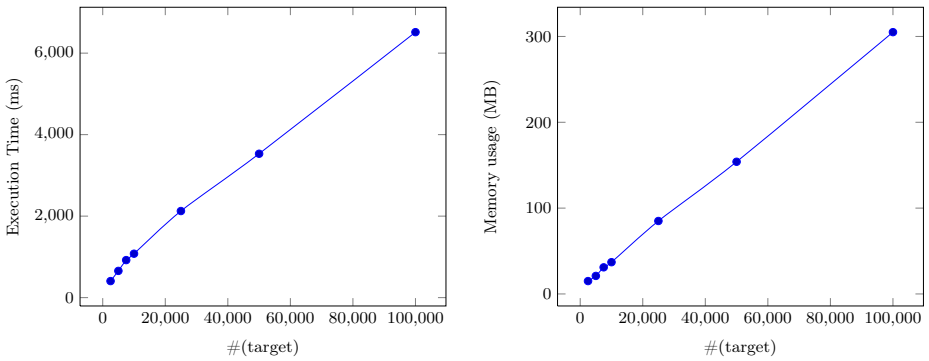

Fig. 9: Varying of the number of tuples in the target relation

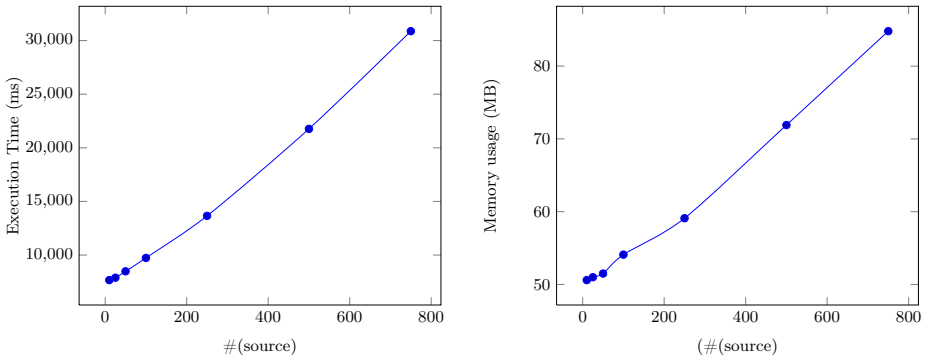

Fig. 10: Varying the number of single attributes in the source schema relation

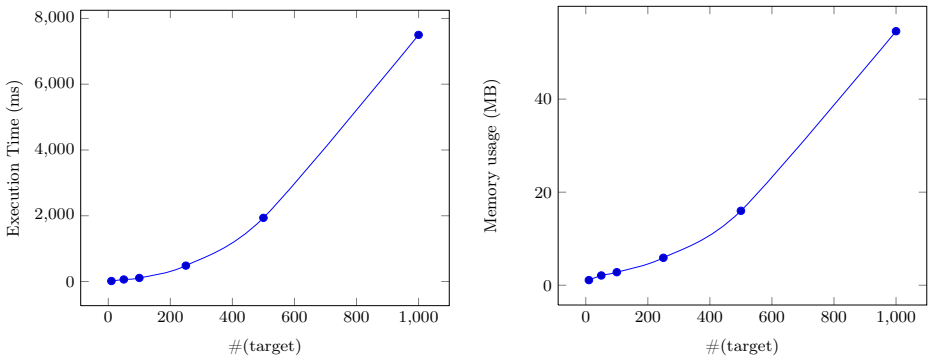

Fig. 11: Varying the number of single attributes in the target schema relation

8 Conclusion

We have addressed the problem of finding an injective mapping between attributes from a source relation to a target relation in an interval-based numerical databases. The proposition is mainly based on the work of [15] on the discovery of approximate unary inclusion dependencies. We implemented the contributions and tested on both real life and synthetic databases. Dealing with intervals instead of classical values turns out to feasible in practice for this polynomial problem, and requires mainly to think about notions such as the membership of an interval into a collection of intervals.
Many perspectives do exist for this work: First, the Astrophysics setting of this paper offers opportunities to extend the contributions made in this paper to real Astrophysics problems. More joint works are needed to better understand the needs of each other. Second, many pattern mining problems can be revisited for interval-based numerical databases. For example, the discovery of conditional functional dependencies could be revisited [6] as well as the discovery of editing rules [7,5] for data cleaning, a main concern in Astrophysics.

Acknowledgments: This work has been partially funded by the CNRS Mastodons projects (QualiSky 2016 and 2017).

References

1. R. K. Ahuja, T. L. Magnanti, and J. B. Orlin. *Network Flows: Theory, Algorithms, and Applications*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1993.
2. D. P. Bertsekas. A distributed algorithm for the assignment problem. 1979.
3. G. B. Dantzig. A history of scientific computing. chapter Origins of the Simplex Method, pages 141–151. ACM, New York, NY, USA, 1990.
4. M. Dawes. The optimal assignment problem.
5. T. Diallo, J.-M. Petit, and S. Servigne. Discovering Editing Rules for Data Cleaning. In *10th International Workshop on Quality in Databases In conjunction with VLDB (Very Large Databases) 2012*, pages 1–8, Aug. 2012.
6. W. Fan, F. Geerts, J. Li, and M. Xiong. Discovering conditional functional dependencies. *IEEE Trans. Knowl. Data Eng.*, 23(5):683–698, 2011.
7. W. Fan, J. Li, S. Ma, N. Tang, and W. Yu. Towards certain fixes with editing rules and master data. *VLDB J.*, 21(2):213–238, 2012.
8. M. J. Golin. Bipartite matching and the hungarian method.
9. S. Khuller. Design and analysis of algorithms: Course notes.
10. M. Klein. A primal method for minimal cost flows with applications to the assignment and transportation problems. *Management Science*, 14(3):205–220, 1967.
11. H. W. Kuhn. The hungarian method for the assignment problem. *Naval Research Logistics Quarterly*, 2:83–97, 1955.
12. H. W. Kuhn. Variants of the hungarian method for assignment problems. *Naval Research Logistics Quarterly*, 3:253–258, 1956.
13. E. Lawler. *Combinatorial Optimization: Networks and Matroids*. Dover Books on Mathematics Series. Dover Publications, 2001.
14. M. Levene and G. Loizou. *A guided tour of relational databases and beyond*. Springer, 1999.

15. F. D. Marchi, S. Lopes, and J. Petit. Efficient algorithms for mining inclusion dependencies. In *Advances in Database Technology - EDBT 2002, 8th International Conference on Extending Database Technology, Prague, Czech Republic, March 25-27, Proceedings*, pages 464–476, 2002.
16. T. Papenbrock, T. Bergmann, M. Finke, J. Zwiener, and F. Naumann. Data profiling with metanome. *PVLDB*, 8(12):1860–1863, 2015.
17. R. T. Snodgrass. The temporal query language tquel. *ACM Trans. Database Syst.*, 12(2):247–298, 1987.