Data Integration and Inconsistencies

Julius Stuller

Institute of Computer Science

Academy of Sciences of the Czech Republic

Bandung, Indonesia, September 2002

- Introduction
- **Inconsistency**
- Integration operations
- **IFAR methodology**
- **Inconsistencies Classification**
- **RIFAR procedure**
- Conclusion

Inconsistency

(A system is said to be consistent if there is no sentence p of the system such that both p and not-p are theorems).

A database has an **inconsistency** *if the data it contains yield under the given interpretation at least one contradiction.*

The interpretation of the data in a database is given by their **semantics** which are, usually – at least partly, stored as *meta-data* in the same database system.

Meta-data present an (axiomatic) theory T (”background knowledge”).
A database has an inconsistency if the data it contains are inconsistent with the theory T, or – in other words – the union of the theory T and of the data contains a contradiction.

Name	Year
Jaromir Jagr	1972
Jaromir Jagr	2001
Mario Lemieux	1965

Without any interpretation we cannot decide at all whether there is or not a contradiction in our database.

First interpretation: *year of the birth*.
Second interpretation: *important year(s)*.

Under the *first interpretation* the given data *yield naturally a contradiction*
(No person can be born in two different years; *consequence:* in this concrete case, at least one datum — year 1972 or 2001— *must be incorrect*).

Second interpretation yields apparently *no contradiction*.

In general the inconsistency says very little about the *correctness* of data.

The concrete data of a given BD which yield a contradiction will be called **inconsistent data**.

Let $\mathcal{B}$ be a database,
Δ the given interpretation of data in $\mathcal{B}$.
We will denote by $\mathcal{I}^{\Delta}(\mathcal{B})$ the inconsistent data of $\mathcal{B}$, or – in case of no possible ambiguity – simply $\mathcal{I}(\mathcal{B})$.

Under our first interpretation the inconsistent data are:

Name	**Year**
Jaromir Jagr	1972
Jaromir Jagr	2001

Integration operations

A1: *The databases to be integrated have no inconsistent data.*

A2: *The DBs to be integrated are relational ones:*

Let $\mathcal{B}_i$ be m relational databases, each consisting of k_i *relations* ${R^i}_j$:

$$ {R^i}_j = \langle\, {A^i}_j\,,\, {D^i}_j\,,\, {T^i}_j \,\rangle\,. $$

From all the usual basic relational operations (and operators) the only ones which can contribute to the process of the integration of databases, and so could lead to possible inconsistencies, are the "update" operations, namely:

- the *unions* of the relations
- the *joins*
  (and the corresponding *compositions*).

The following relational operations:

- the *unions* of the relations
- the *(equi -) joins*
- the *(equi -) compositions*

will be called the **integration operations**.

We will use the symbol $\int$ to denote any integration operation without specifying exactly if it is an union, a join or a composition.

We will use the notation $\int_{i=1}^{m} \mathcal{B}_i$ to denote the integration of databases $\mathcal{B}_i$ without specifying explicitly what integration operation(s) were/are/will be used on the appropriate relations $R^i{}_j$.

Union of the Relations

In order to be able to make the union of the relations ${R^{i_j}}_{q_j}$ we must first suppose they all have the same degree, say k :

A3: $(\exists k \geq 1)\,(\exists s \geq 2)\,(\forall j \in \widehat{s})\,(\exists \mathcal{B}_{i_j})$
$(\exists\, {R^{i_j}}_{q_j} \in \mathcal{B}_{i_j})\,(|\, {A^{i_j}}_{q_j} \,| = k)$

We can always find, by successive projections, the corresponding subrelations (of some ${R^{i_j}}_{q_j}$) with the required property.

Furthermore, for simplification, we will suppose the relations ${R^{i_j}}_{q_j}$ are defined over the same relational schema $\mathcal{S}$:

A4: $(\forall j \in \widehat{s})\,({R^{i_j}}_{q_j} \sqsubset \mathcal{S} = \langle\, A\,,\, D\,\rangle)$

R_1	
Name	**Position**
Jordan	player

R_2	
Name	**Position**
Jordan	owner

$R = R_1 \cup R_2$	
Name	**Position**
Jordan	player
Jordan	owner

Functional dependency : **Name** $\rightarrow$ **Position**

The data of the database $\mathcal{B}$ not satisfying the given set of the integrity constraints Σ will be denoted by $\mathcal{I}_{\Sigma}(\mathcal{B})$ and called:
the **inconsistent data** *with respect to the set of the integrity constraints* Σ .

In general the following inclusion holds:

$$\mathcal{I}_{\Sigma}(\mathcal{B}) \subset \mathcal{I}^{\Delta}(\mathcal{B})$$

More we are able to *describe precisely the semantics of data* (and by this also *their interpretation*) in the form of the *appropriate integrity constraints* (and our *database system* **must** *be able to process* **all** *of them*), **more** we can *expect to automatize the process of discovering the inconsistencies* in the integration of databases.

The **ideal** *situation is the one in which we can consider the given set of integrity constraints as completely describing the semantics of data*:

A database instance r *is consistent if* r *satisfies* IC – the given set of integrity constraints – *in the standard model-theoretic sense, that is* $r \models IC$; r *is inconsistent otherwise*.

In such a (ideal) case the following equality holds:

$$\mathcal{I}^{\Delta}(\mathcal{B}) = \mathcal{I}_{\Sigma}(\mathcal{B})$$

The *contrary naturally leads* to a greater extent of *manual procedures*.

In recent years there have been proposed some heuristics for searching of inconsistencies (see e.g. [Castro & Zurita (1998)]).

Returning again to our example:

R_1	
Name	**Position**
Jordan	player

R_2	
Name	**Position**
Jordan	owner

$R = R_1 \cup R_2$	
Name	**Position**
Jordan	player
Jordan	owner

Functional dependency : **Name** $\rightarrow$ **Position**

We can see that the inconsistent data (with respect to the given set of the integrity constraints) of the integrated database are equal to the *whole integrated database*.

Our final **goal** is to *minimize the inconsistencies* in the integrated database or, in other words, to *minimize the inconsistent data*.

Naturally, the appropriate integrity constraints can largely help us in this and so we will always start by minimizing the inconsistent data *with respect to the given set of the integrity constraints*.

Unfortunately the real situations (specially in the case of the **Web data**) may be much more complicated as the required helpful integrity constraints are very often *incomplete* or even *missing* completely ...

The IFAR Methodology

Step 1: **I***ntegrate* databases $\mathcal{B}_k$: $\int_{k=1}^{m} \mathcal{B}_k$

Step 2: **F***ind* the set of inconsistent data:

$$\mathcal{I}(\int_{k=1}^{m} \mathcal{B}_k)$$

Step 3: **A***nalyze* the set $\mathcal{I}(\int_{k=1}^{m} \mathcal{B}_k)$ in order to find:

- *Inconsistent data with respect to the given set of the integrity constraints* Σ :

  $$\mathcal{I}_{\Sigma}(\int_{k=1}^{m} \mathcal{B}_k)$$

  $(\exists\, i \in \widehat{m})\ (\exists\, j \in \widehat{k}_i)\ (\exists\, R^i{}_j = \langle\, A^i{}_j, D^i{}_j, T^i{}_j \,\rangle\,)$
  $(\exists\, t \in T^i{}_j\,)\ (\, t \not\models \Sigma\,)$

  Such a t may not represent correctly a fact from the reality we are trying to capture in a database – in the relation $R^i{}_j$

  (In our example case it could mean that either Jordan is not a *player* or that he is not an *owner*.)

- *Wrong integrity constraints*:

  Some of $\mathcal{I}_\Sigma(\int_{k=1}^{m}\mathcal{B}_k)$ being correct could imply some integrity constraints from Σ may be wrong – they may not correctly reflect the reality we are trying to model

  (In our example it could mean that there may be *more than one Position associated with one Name*.)

- *Wrong descriptions* of data:

  Some of $\mathcal{I}_\Sigma(\int_{k=1}^{m}\mathcal{B}_k)$ being correct could imply some attributes (description) are wrong

  (In our Example 3 it could mean, for instance, that datum "owner" is not a – value of the attribute – *Position*, but it should be a – value from yet an other attribute – *Function*.)

Step 4: **R***esolution* of the inconsistencies:

- *"Correction of data"*: *New relations* $\widetilde{R^i}_j$ (without *incorrect − wrong − data*) over which we will do *integration* $\int_{i,j} \widetilde{R^i}_j$. The incorrect data should be discovered and corrected at the *data integration* stage.

- *"Correction of integrity constraints"*: *New set of integrity constraints* $\widetilde{\Sigma}$ (without *wrong integrity constraints*). (At least some of) the wrong constraints should be discovered and their correction should be performed already at the *schema integration* stage.

- *"Correction of attributes"*: *Renaming* of the *wrong attributes*. (It should be done only after a thorough − *semantical* − analysis of data corresponding to the incorrect attributes.) (Some of) these incorrect attributes should be discovered and their renaming should be performed again at the *schema integration* stage.

Π - Unions

Next we will suppose the relations $R^{i_j}{}_{q_j}$ are defined over such different relational schemata $\mathcal{S}^{i_j}{}_{q_j} = \langle\, A^{i_j}{}_{q_j}\,,\, D^{i_j}{}_{q_j}\,\rangle$ that there exist appropriate permutations $\pi^{i_j}{}_{q_j}$ in $|\,\widehat{A^{i_j}{}_{q_j}}\,|$ that the following holds:

A5: $$\bigcap_{j=1}^{s} D^{i_j}{}_{q_j}\,(\,\pi^{i_j}{}_{q_j}\,(\,A^{i_j}{}_{q_j}\,)\,) \neq \emptyset$$

R_1	
Name	**Position**
Lemieux	player

R_2	
Name	**Function**
Lemieux	owner

$R \;=\; R_1 \cup_\pi R_2$	
Name	**Post**
Lemieux	player
Lemieux	owner

We presuppose the (names of the) attributes **Position** and **Function** are synonyms (i.e. they are semantically equivalent).

Relaxing the condition **A4** (about the relations one wants to make an union over being defined over the same relational schema) into weaker condition **A5** requiring the existence of permutations ${\pi^{i_j}}_{q_j}$ such that there exists the π - union of relations ${R^{i_j}}_{q_j}$, one can obtain by similar reasoning we used to the union of relations the same sources of possible inconsistencies:

- *Inconsistent data with respect to the given set of the integrity constraints*
- *Wrong integrity constraints*
- *Wrong descriptions* of data.

and so the **IFAR** *methodology* can be used again.

(Equi -) Joins

Difference between the *integration* by:

- one of the *joins* (the natural one)
- one of the *unions* (the π - *union*)

R_1	
Mother	**Son**
Eve	John

R_2	
Mother	**Daughter**
Eve	Anne

$R = R_1 * R_2$		
Mother	**Son**	**Daughter**
Eve	John	Anne

$R = R_1 \cup_\pi R_2$	
Mother	**Child**
Eve	John
Eve	Anne

Depending on the every concrete situation one must choose the *best appropriate operation* to perform the integration of the databases.

For instance, in a case of a **data warehouse** , from the point of view of *data mining* techniques, the *integration by (natural) join* will be very *probably preferred*.

In case of **incomplete information**, specially *missing values*, the usage of the **outer-join** (for instance *left* or *right*) may be useful ...

R_1	
Husband	**Wife**
Joseph	Mary

R_2	
Mother	**Child**
Mary	Jesus

$R = R_1 *_{Wife=Mother} R_2$		
Husband	**Wife**	**Child**
Joseph	Mary	Jesus

Again, as in the case of the union, even in this very simple example, without any further supplementary information it is impossible to decide whether an inconsistency appeared in the process of the integration of databases.
The comparison of this join with the π - union of the same relations:

$R = R_1 \cup_\pi R_2$	
Man	**Woman**
Jesus	Mary
Joseph	Mary

shows that the integration by joins against the integration by unions:

- allows **new** *relationships between objects* (entities or their attributes, and this is exactly what is usually one looking for in any **data mining** technique), which
- can be the source of **new** *inconsistencies* (having for arguments some of such new relationships) in addition to the inconsistencies known from the unions.

In any case the **IFAR** *methodology* can be used again.

Condition on p relations $R^{i_k}{}_{q_k}$ to be joinable

A6: $$\bigcap_{k=1}^{p} D^{i_k}{}_{q_k} (\pi^{i_k}{}_{q_k} (B^{i_k}{}_{q_k})) \neq \emptyset$$

where $(\forall k \in \widehat{p}) (B^{i_k}{}_{q_k} \subset A^{i_k}{}_{q_k})$

which is equal to the condition **A5** with a unique difference that $B^{i_k}{}_{q_k} \subset A^{i_k}{}_{q_k}$ and so one can have in principe up to

$$\prod_{k=1}^{p} (\sum_{m=1}^{|B^{i_k}{}_{q_k}|} \binom{|A^{i_k}{}_{q_k}|}{m})$$

possibilities of performing the join of p relations.

Inconsistencies classification

A7: *Let* $m \geq 2$,
B_k *be* m *DBs one wants to integrate,*
Σ_k *be* m *corresponding sets of ICs,*
and Σ_{m+1} *be the set of the ICs corresponding to the result of database integration operation* $\int_{k=1}^{m} \mathcal{B}_k$
such that $\Sigma = \bigwedge_{k=1}^{m+1} \Sigma_k$ is (logically) consistent.

Let $\mathcal{B}_k$ be m databases satisfying **A7**.
We will call any inconsistencies in the result of the database integration $\int_{k=1}^{m} \mathcal{B}_k$
the **data integration inconsistencies**, specially:

- **universe of discourse inconsistencies**
$$\Leftrightarrow \ (\exists\, k \in \widehat{m}\,)\ (\exists\, \widetilde{A^i}_k \neq A^i{}_k\,)$$
- **data inconsistencies**
$$\Leftrightarrow \ (\exists\, k \in \widehat{m}\,)\ (\exists\, \widetilde{R^i}_k \neq R^i{}_k\,)$$
- **integrity constraints inconsistencies**
$$\Leftrightarrow \ (\exists\, k \in \widehat{m+1}\,)\ (\widetilde{\Sigma_k} \neq \Sigma_k\,)$$
- **semantical inconsistencies**
$$\Leftrightarrow \ (\exists\, k \in \widehat{m}\,)\ (\exists\, \pi^i{}_k \neq Identity\,)$$

($\widetilde{A}$ being a subset of the set A containing no wrong attributes).

We will call data integration inconsistencies shortly the **integration inconsistencies**.

The *universe of discourse inconsistencies* and the *integrity constraints inconsistencies* will be called the **conceptual inconsistencies**.

Every type of the integration inconsistencies originates from *different* **sources** and therefore *can be best* **eliminated**, or
at least **minimized**, at *different* **stages** of the integration of the concerned databases:

- the *conceptual inconsistencies* at the stage of the **schema integration**
- the *semantical inconsistencies* by **well-considered choice** of the *attribute(s) over which one wants to integrate the DBs* (maybe for the purpose of the *envisaged* **data mining** in a given data warehouse)
- the *data inconsistencies* by *thorough* **verification** and **validation**, at the *data entry* stage, and
  data cleansing at subsequent stages.

In the following we will use the notation:

δ - **inconsistencies** :
database integration inconsistencies

u - inconsistencies :
universe of discourse inconsistencies

d - inconsistencies :
data inconsistencies

i - inconsistencies :
integrity constraints inconsistencies

s - inconsistencies :
semantical inconsistencies

c - inconsistencies :
conceptual inconsistencies.

In order to eliminate, as much as possible, the occurrences of integration inconsistencies one should try to, especially in the case of the validity of the conditions **A1 & A2 & A7 &**

- **A4:** *clear the DBs* to be integrated from:
  - **wrong data** which *can lead to the* **d** - *inconsistencies*
  - **wrong integrity constraints** which *can lead to the* **i** - *inconsistencies*
  - **wrong attributes** which *can lead to the* **u** - *inconsistencies*
- **A5:** *semantically* **deeply analyze** the *corresponding attributes* in the relations to be integrated by π - **unions** to eliminate the **s** - *inconsistencies*
- **A6:** *semantically* **deeply analyze** the *corresponding attributes* in the relations to be integrated by **joins** to eliminate the **s** - *inconsistencies*.

The RIFAR procedure

Step 0: **R***esolve the conflicts in* $\Sigma = \bigwedge_{k=1}^{m+1} \Sigma_k$

Put $i = 1$

Step 1: *While* $i < m-1$: *Put* $i = i+1$

I*ntegrate the DB* $\mathcal{B}_i$ *with* $\int_{k=1}^{i-1} \mathcal{B}_k$

Put $j = 0$

Substep 1A: *While* $j < (k_i - 1)$: *Put* $j = j+1$

Integrate the relation $R^i{}_j$ *with*

$\int_{s=1}^{j-1} R^i{}_s \int_{k=1}^{i-1} \mathcal{B}_k$

Subsubstep 1A2: **F***or every tuple* t *from* $R^i{}_j$
verify if it does lead to an inconsistency
(with respect to the given set of the ICs Σ_{m+1}*)*

Subsubsubstep 1A2a: *If it does :*

- *remove the corresponding tuple(s) from* $\int_{s=1}^{j-1} R^i{}_s \int_{k=1}^{i-1} \mathcal{B}_k$ *if this does not violate* Σ_{m+1} ,
  otherwise make a copy of it/them
- *put it/them together with* t *into* $\mathcal{I}(\int_{k=1}^{m} \mathcal{B}_k)$
- *index them all by the corresponding IC(s)*

Subsubsubstep 1A2b: *If it does not, integrate it with* $\int_{s=1}^{j-1} R^i{}_s \int_{k=1}^{i-1} \mathcal{B}_k$

Step 3: **A***nalyze the set* $\mathcal{I}(\int_{k=1}^{m} \mathcal{B}_k)$ *by*:

Substep 3A: *Decomposing it into subsets indexed by the set(s)* Q *of the same integrity constraint(s)* $\mathcal{I}(\int_{k=1}^{m} \mathcal{B}_k)_Q$ *to find*:

- $\mathcal{I}_{\Sigma_{m+1}}(\mathcal{B})$
- *wrong integrity constraints*
- *wrong descriptions* of data

Step 4: **R***esolution of inconsistent and wrong items:*

- *correction of data* (in order to obtain $\widetilde{R^i}_j$)
- *correction of ICs* (in order to obtain $\widetilde{\Sigma}_i$)
- *correction of attributes* (in order to obtain $\widetilde{A^i}_j$).

Conclusions

The *occurrence of certain types of data inconsistencies can provide an useful* **feedback** to, for instance, the **conceptual modelling** of a *data warehouse* (to its *logical schema design*), to a more *intelligent data entry*, *data verification* and *validation*, and to a possible *better selection* of the appropriate *data mining techniques / methods*.
For instance, the occurrence of any of **c - inconsistencies** can trigger a positive feedback to the conceptual modelling of a concrete data warehouse as, depending on its precise type, it can either signal wrong attribute(s) existence in the case of the **u - inconsistencies**, either wrong integrity constraint(s) existence in the case of the **i - inconsistencies**.

As in the case of relations in the relational data model the semantics (metadata) of data (description of attributes, corresponding integrity constraints, etc.) are – or can be – stored in the same type of relations (called *system* relations), our **IFAR** *methodology* and **RIFAR** *procedure* can be applied to any conflict of similar schema structures:

- (*value–to–value conflicts*,
- *attribute–to–attribute conflicts* and
- *table–to–table conflicts*)

on the one side, but also to any conflicts of different schema structures:

- (*value–to–attribute conflicts*,
- *value–to–table conflicts* and
- *attribute–to–table conflicts*)

on the other side.

The ideas presented here (**RIFAR** *procedure*) have been partly implemented in a prototype system to provide *support for the resolution of the inconsistencies in the process of the integration of databases*.

In the future we would like to further elaborate our *methodology* and *procedure* by incorporating it into an intelligent agent system and taking more advantage of the soft computing paradigm.