USING INFORMATION CAPACITY TO ASSESS INTEGRATED SCHEMA

(Complete)

Andrea Maurino
University of Milano Bicocca Italy
maurino@disco.unimib.it

Carlo Batini
University of Milano Bicocca Italy
batini@disco.unimib.it

Simone Grega
NexTTLab srl, Italy
simone.grega@nexttlab.it

Abstract: In large organizations the database architecture is typically built through a series of projects and realizations that result in a number of heterogeneous and overlapping data sources. This trend is worsened by merger and acquisition activities that add new data sources from external organizations to the existing data architecture. Data fragmentation significantly reduces the possibility for an organization to exploit its information assets. A technology that partially alleviates these problems is data integration middleware that allows users to read-only access data stored in heterogeneous data sources through the presentation of a unified view of these data. In this paper we introduce a new concept to measure the improvement in information capacity enabled by data integration solutions, and propose an original framework which can support the evolution of the organization data architecture by identifying the optimal solution that maximizes such improvement within a given cost threshold.

Key Words: Information capacity, Cooperative information systems, Data integration middleware, Data architecture.

INTRODUCTION

Organizations tend to create databases of interest through a series of projects and realizations that result in a database architecture characterized by a set of anomalous behaviors. With the term data architecture we define the allocation of the data of interest to an organization among the (usually many) database management systems available in the organization's information system. The above mentioned behaviors concern the redundancy of representations, the misalignment of data among different databases, the scarce coherence in business rules related to the same objects in different databases, and errors in data that result in the heterogeneous representations of records pertaining to the same real world object. This trend is made more and more critical by the continuous evolution of organizations due to merger and acquisition activities. Consequently the problem of managing the whole data architecture migrating from traditional DBMS technologies to integration technologies is a primary issue in modern organizations.

Among existing solutions on the market [7], data integration is the most promising middleware for all cases where there is the need to allow users to read-only access data stored in heterogeneous data sources through the presentation of a unified view of these data. In the last few years, both industry and academia have investigated data integration solutions both from theoretical and practical view points (see [6] for a survey). Two main approaches to data integration can be identified, based on the actual location of data stored in sources to be integrated. In virtual data integration the unified view is virtual, and data reside only at sources. A reference architecture for virtual data integration middleware is the mediator-wrapper architecture [15]. The second approach, namely materialized data integration, provides a (unified view of) data that is materialized in a data warehouse [12]. In the case where up-to-date data are needed and the

periodical update of the materialized view is costly, virtual data integration is preferred to data warehouse.
In this paper we focus on data integration technologies, discarding the analysis of data warehouse solutions. Even if a lot of results have been obtained in the field of data integration, to the best of our knowledge the investigation of benefits related to extended access to integrated data made possible by data integration, has not be investigated so far in comparison to costs of such a solution. In this paper we propose a suite of methods for the evolution of the data architecture of an organization, with the goal of optimizing the quality of the overall data architecture when data integration solutions are used. The quality of the data architecture is measured in terms of a new concept, which we introduce in the paper, its *potential information capacity*. The potential information capacity can be roughly defined as the set of all types of data that can be extracted from a (virtually) integrated database schema that cannot be extracted considering non integrated data sources alone. We face the problem considering first the case of the adoption of one single data integration solution that leads to a unique virtual schema, and then considering the adoption of n > 1 data integration solutions on clusters of schemas. Furthermore, we establish a constraint on the cost of the solution. An algorithm based on a branch and bound technique is proposed, working for a number of schemas that does not exceed 30-50 and a number of concepts that does not exceed 100-150.

The paper is organized as follows. In Section *Running Case* we describe the example that will be used in the paper. In Section *Preliminary Definitions* we introduce the graph model used to describe the schemas. Section *Information Capacity* presents the formal definition of the potential information capacity. Section *Integration Costs* introduces the cost model we adopt, and in Section *Choice Of The Optimal Architecture(S)* two different algorithms for the selection of optimal data architecture solutions, in case, respectively, n = 1 and n > 1 integration solutions are described. Section *Related Works* discusses the state of the art, and Section *Conclusion* discusses future research work.

RUNNING CASE

The running case deals with a company that produces and sells various types of goods on the market. The company has organized its data of interest in six different databases, namely: *Production-items*, *Sales*, *Organizational structure- Human Resources*, *Production process*, *Organizational structure* - *Contracts* - *Salaries*, *Clients*.
In the paper and in the case study we assume that the schema resulting from the integration of the whole set of schemas is known. This assumption is reasonable in the case study and in organizations where the total number of schemas does not exceed the threshold defined in the introduction, say, 30-50 schemas. The schema resulting from the integration of the six conceptual schemas is shown in Figure 1 where the six source schemas are surrounded by surfaces with different shapes. The model used for describing conceptual schemas is the Entity Relationship model enriched with generalization hierarchies [2].
The diagrammatic representation uses simple names in bold characters instead of boxes for entities, lines with names in the middle for relationships, arrows for IS-A relationships and generalizations, minimum and maximum cardinalities among parenthesis, and for reasons of simplicity, does not include attributes.
We also assume absence of synonyms and homonyms for names of entities. In Figure 2 a table that for each pair of schemas (*Si, Sj*) lists common concepts among *Si* and *Sj* is shown.

PRELIMINARY DEFINITIONS

In the following, to be able to formally manage the framework we use a graph-based model instead of the ER model.

Definition 1 - Let $G = (N,A,R, I, T, E,C, f)$ be a graph representing an ER schema Φ, where:

- N is the set of nodes of an oriented graph representing the entities of Φ,

- *A* is the set of nodes of an oriented graph representing the attributes of *Φ*,
- *R* is the set of oriented edges, representing the relationship between entities of *Φ*,
- *I* is the set of oriented edges, representing the is-a hierarchies among entities of *Φ*,
- *T* is the set of oriented edges, representing the specialization hierarchies among entities of *Φ*,
- *E* is the set of edges between a node $n \in N$ and a node $a \in A$,
- *C* is the set of labels *{*"0..1", "1..1", "1..*N*", "0..*N*"*}*,
- *f* is a function associating a value of *C* to edges $r \in R$, representing minimum and maximum cardinalities.

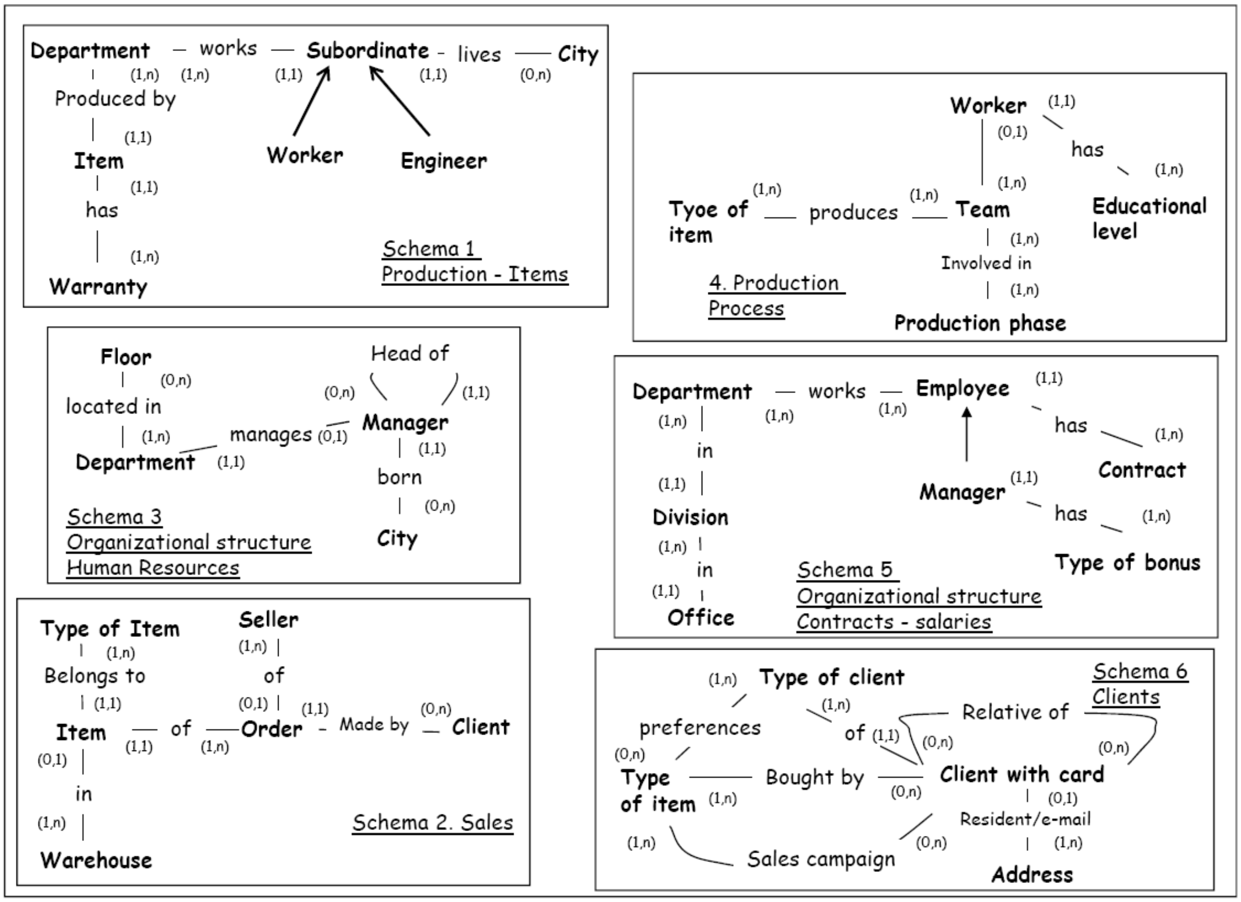


Figure 1 The integrated schemas with input schema

It is possible to associate a *Naritec* graph ("Naritec" is the word composed of the first seven sets of the above mentioned definition) to any ER schema. For example, given the schema of Figure 3a including five entities, five relationships and one is-a hierarchy, the associated Naritec graph is shown in Figure 3b, where continuous circles represent nodes *n* in N (that is, entities of the schema), dotted circles represent attributes, continued lines represent relationship edges, pointed lines represent IS-A hierarchies, dashed dotted lines represent specialization hierarchies, and dashed lines represent connections between entities and relationships. According to the semantics of the Naritec graph, given an ER schema *Φ*, there exists exactly one Naritec graph associated to it.

Schemas and common concepts	**1. Production - Items**	**2. Sales**	**3. Org. Structure Human Resources**	**4. Production process**	**5. Org. Structure Contracts Salaries**	**6. Clients**
1. Production - Items	-	Item	City Department	Worker Item	Department Manager	Item
2. Sales	Item	-	-	Item Type of item	-	Item Type of item
3. Org. Structure Human Resources	City Department	-	-	-	Department Manager	-
4. Production process	Worker Item	Item Type of item	-	-	-	
5. Org. Structure Contracts Salaries	Department Manager	-	Department Manager	-	**-**	**-**
6. Clients	Item	Item Type of item	-	Item Type of item	**-**	**-**

Figure 2 Common concepts among schemas

Definition 2 - An *SQL query Qi* over a schema *Φ* is any subgraph of the corresponding Naritec graph corresponding to the set of nodes *n* (that is the entities listed in the FROM clause), a set of attributes *a* (that is the attributes in the SELECT clause), a set of edges *a* and a set of edges *r* (that is the join path between two entities), *t* and *i*.

A *relationship path* $L_{n_i}^{n_j}$ is a path over a Naritec graph composed by edges $r \in R$, $i \in I$, and $t \in T$, starting form the node n_i and ending in the node n_j

Definition 3 - A *relationship path* is defined as $L_{n_i}^{n_j}$= *{l*1, *l*2, .., *lz}*, $li \in R \vee I \vee T$. With $|L_{n_i}^{n_j}|$ we represent the cardinality of $L_{n_i}^{n_j}$, that is the number of edges belonging to the path.

Notice that there are many relationship paths with the same starting and ending entities.
The Join Adjacency Matrix is the adjacency matrix of the naritec graph representing the join path between entities.

Definition 4 - A *Join Adjacency Matrix* (JAM) is a |*N*|*x*|*N*| matrix, where:

- |*N*| is the number of nodes *n* of a Naritec graph associated to schema *Φ*,
- the diagonal entry $JAM_{ii}$is twice the number of loop edges over entity i

the non-diagonal entry JAM_{ij} is the number of edges *r* connecting node *i* to node *j* and edges *r* connecting a node k to node j iff node k has an incoming edge *i* exiting from node i.

The above definitions will be used in the following, referring to the concept of information capacity and the related algorithms for the choice of the optimal architecture..

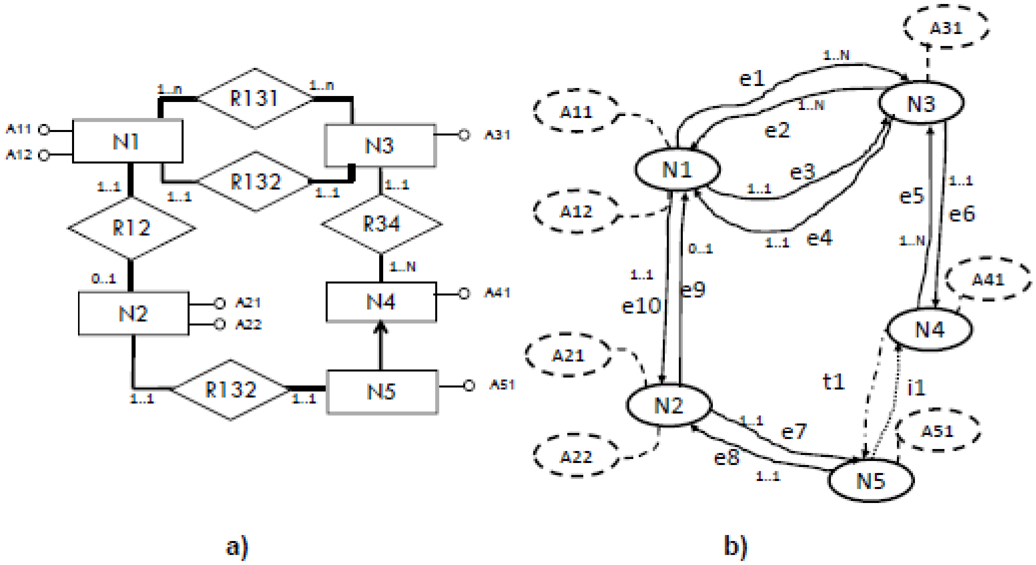


Figure 3 Example of Entity Relationship schema and associated *naritec* graph

INFORMATION CAPACITY

In this section we formally define three types of information capacity that capture from three different points of view the intuitive concept of information content of a database schema or a set of database schemas using a data integration solution; they are: schema based information capacity, instance based information capacity, and potential information capacity.

Schema based information capacity

In order to formally express the intuitive concept of ”information content that can be extracted from a database”, the first definition of information capacity considers only the conceptual schema of the database, discarding the database extension, namely the data. We introduce the schema based information capacity IC^{sb} in an iterative way, starting from the general formula:

$$IC(\phi) = \sum(Q^i) \tag{1}$$

where the generic Q_i is a query involving i nodes *n* connected by means of $i - 1$ edges *r*,*i*, or *t* of a Naritec graph[1]. In our approach two queries are considered different if and only if they involve either at least one different entity or a different relationship Notice that we do not consider predicates (which affect at instance level only) nor attributes. It is straightforward to show that the number of queries involving exactly one entity (Q_1) is equal to |*N*|, where |*N*| is the number of nodes N of a Naritec graph representing the entities of the schema *Φ*. The number of queries involving exactly two entities (Q_2) is equal to the number of all paths of length equal to 1 of a Naritec graph. Generalizing, the number of queries involving exactly k entities (Q_k) is equal to the number of paths of length k-1. The number of queries involving k (*k* > 1) entities is defined as:

[1] We do not consider the case of relationships connecting more than two entities that are very infrequent in real cases

$$Q^k = \sum_{ij} jam_{i,j}^k \qquad (2)$$

where $jam\,_{i,j}^{k}$ is the i,j cell of jam matrix elevated to k

The result of raising the Join Adjacency Matrix to the k-th power produces a new matrix whose generic cell $jam\,_{i,j}^{k}$ represents the number of paths of length k from i to j.

Definition 5 - The *schema based information capacity* (IC^{sb}) associated to a schema s is expressed by the formula

$$IC^{sb}(\Phi) = \sum Q_i \;=\; |N| + \sum_{k=1..|N|} \sum_{i,j\,\in|N|} jam_{i,j}^k \qquad (3)$$

Since it is quite unusual that a query may involve, say, more than three or four entities, not all queries with an arbitrary number of involved entities should be considered in the evaluation of the IC^{sb}. Consequently, equation (3) can be rewritten as follows:

$$IC^{sb}(\Phi) = \sum Q_i \;=\; |N| + \sum_{k=1..\lambda} \sum_{i,j\,\in|N|} jam_{i,j}^k \qquad (4)$$

Where λ is the maximum number of involved entities in a query. Consider now the ER schema and the associated Naritec graph of Figure 3b), the associated JAM is shown in Table 1, and let $\lambda = 4$; it results that the IC^{sb} of the Naritec graph of Figure 3b) is equal to 143. In fact, there are five entities plus 12 paths of length one (including two entities), 34 paths of length two (including three entities), 92 paths of length three (including four entities).

	$N1$	$N2$	$N3$	$N4$	$N5$
$N1$	0	1	2	0	0
$N2$	1	0	0	0	1
$N3$	2	0	0	1	1
$N4$	0	0	1	0	0
$N5$	0	1	1	0	0

Table 1 JAM for the Naritec graph of Figure 3

Instance Based Information Capacity

While the schema based information capacity is a useful and easy way to evaluate index to measure the information that can be extracted from a schema, another possible point of view is to also consider the instances of the database. For these purposes, we enrich the above mentioned graph definition by introducing two new sets $W \in R^+$ and $P \in [0..1]$ and a function $wp : x, y \in R^+ \rightarrow e \in E$. Let $e \in E$ be an edge derived from relationship r connecting two entities n_i and n_j , $w \in W$ represent the average number of instances of n_i participating in the relationship, and $p_i \in P$ describes the probability that an instance of entity n_i participates in the relationship r. Notice that the average number of instances participating in a relationship and the probability values can be calculated by using one of the techniques described in [11] or by using the statistical functions available in DBMSs [1]. Figure 4 shows the extended version of the Naritec graph (Naritec+ in the following) of Figure 3. IC^{ib} is the amount of instances that is possible to retrieve by means of all possible relevant queries. Formally it is defined as follows:

Definition 6 - Assume that Q_i^i represents the number of instances extracted by queries identified in the previous section. The *instance based information capacity* IC^{ib} is

$$IC^{ib}(\Phi) = \sum Q_i^i \qquad (5)$$

The number of instances retrieved by Q_i^1 queries, that is considering queries involving exactly one entity, is equal to $||n_i||$, that is the sum of the instances of all entities of the schema Φ. The number of instances retrieved by the generic Q_i^k query is defined as follows:

$$Q_i^k = ||n_i|| * \sum_{z \,\in\, L_{n_i}^{n_j}} (w_z * p_z) \qquad (6)$$

where $||n_i||$ is the number of instances of entity n_i, w_z, p_z are the weight and the probability associated to an edge of a relationship path $L_{n_i}^{n_j}$. Thus we can define IC^{ib} as:

$$IC^{ib}(\Phi) = \sum Q_i^i = \sum ||n_i|| + \sum \left||n_i|\right| * \sum_{z \in L_{n_i}^{n_j}} (w_z * p_z) \qquad (7)$$

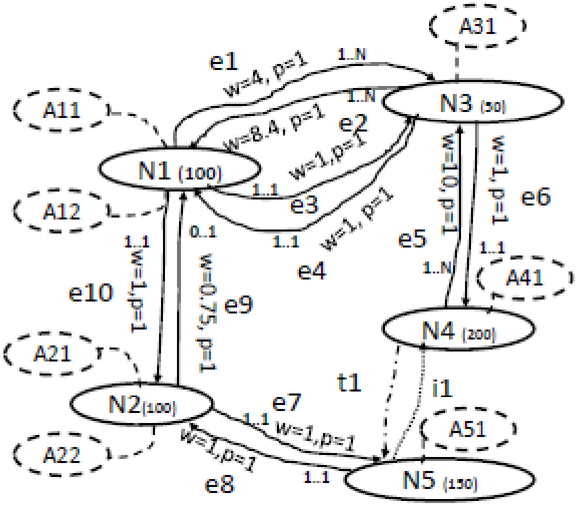
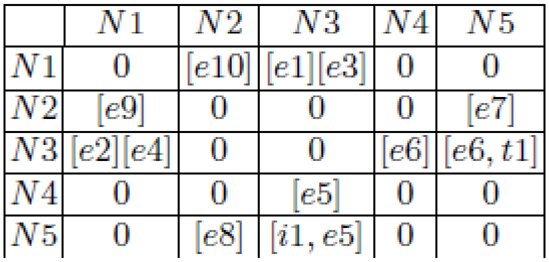


Figure 4 Naritec+ graph of Figure 3

For example, consider the schema of Figure 3, the entities N_1,N_3 and the edges e_1, e_2; furthermore, assume that
– $||N_1||$= 100, $||N_3||$= 50,
– w_{e_1}= 4, p_{e_1} = 1
– w_{e_2} = 8.4, p_{e_2} = 1
In this case, there are four possible relationship paths of maximum length 2, over the above described subschema, namely, the schema formed by nodes [e_1], [e_2] and edges [e_1, e_2], [e_2, e_1]. The instance based information capacity is equal to IC^{ib} = 100 + 50 + 100 * 4 * 1 + 50 * 8.4 * 1 = 970.
In order to calculate IC^{ib} over a generic Naritec+ graph we have to identify all possible paths in the graph. To solve this problem, we consider the JAM matrix again. In particular, in order to identify all paths of length N, JAM is elevated to N, but, instead of just counting the paths (as described previously), the algorithm we propose lists them, by modifying the adjacency matrix so that the (i,j) entry of the matrix is a list of relationship paths from i to j. The matrix multiplication algorithm is modified so that instead of multiplying and adding cell values in the matrix multiplication algorithm, it concatenates cell values. Let us consider the example of Figure 3, the modified JAM matrix G is described in Table 2.



	$N1$	$N2$	$N3$	$N4$	$N5$
$N1$	0	$[e10]$	$[e1][e3]$	0	0
$N2$	$[e9]$	0	0	0	$[e7]$
$N3$	$[e2][e4]$	0	0	$[e6]$	$[e6, t1]$
$N4$	0	0	$[e5]$	0	0
$N5$	0	$[e8]$	$[i1, e5]$	0	0

Table 2 JAM for the naritec+ graph of Figure 3

For example, by invoking the procedure *Pow*(G, 2), all possible paths from node $N1$ to $N1$ are [e_{10}, e_9], [e_1, e_2], [e_1, e_4], [e_3, e_2], [e_3, e_4],

Potential Information Capacity

The potential information capacity (IC) is defined for schemas resulting from integration of other schemas. Intuitively, the (schema based/instance based) potential IS of an integrated schema resulting from the integration of a set of schemas S_1, S_2, S_n, is the additional (schema based/instance based) IC of the integrated schema that can be exploited with respect to the sum of the (schema based/instance based) ICs of single schemas. More formal definitions have to be specialized according to the knowledge available on schemas.

Schema based Potential Information Capacity

The *schema based potential information capacity* of the integrated schema composed of two schema *A* and *B* is defined as: $PIC\ ^{sb}_{A,B} = IC^{sb}_{A,B} - IC^{sb}_{A} - IC^{sb}_{B}$

The schema based potential information capacity of an integrated schema is the set of join paths associated to the integrated schema that are new w.r.t. the set of join paths of the local schemas, namely, the IC of the integrated schemas minus the overall IC of local schemas individually considered

For example, consider the S_1 (Production-Item) and S_2 (Sales) schemas of Section *Running Case*. The two schemas share the same entity *Item*. The schema based potential IC (with a maximum join path length equal to four) of schema S_1 is $IC^{sb}_{S_1}$ = 438, while the IC of schema S_2 is $IC^{sb}_{S_2}$ = 160.

The schema based IC of the integrated schema is $IC^{sb}_{S_{1,2}}$ = 844, thus the schema based potential information capacity is P$IC^{sb}_{S_{1,2}}$= 844−438−160 = 246, representing 30% of additional queries that cannot be extracted in a data architecture that does not make use of data integration solutions.

Instance based potential information capacity

The instance based potential information capacity is defined in the same way as the schema based one, that is $PIC^{ib}_{AB} = IC^{ib}_{A,B} - IC^{ib}_{A} - IC^{ib}_{B}$. An important aspect that differentiates PIC^{ib} with respect to PIC^{sb} is the consideration of instances of common entities stored in different schemas. In an information system where data are affected by errors or heterogeneities of various types, integration activities have to be performed to solve instance level heterogeneities, reconciling the different data values related to the same entity instance in the real world [4]. The activity called *record linkage* aims to cluster all the tuples referring to the same entity instance of the real world, notwithstanding the possible presence of errors or heterogeneities in tuples. Consequently, given an entity *m* belonging to two schemas S_1 and S_2, the number of instances in common among the two databases is a value $0 \le ||m_{S_1 S_2}|| \le min(|| m_{S_1}||,|| m_{S_2}||)$. It is worth noting that the record linkage activity is a complex and time/cost consuming task [4] and in some cases, it is impossible to perform, as in the case of an inter-organizational data integration architecture.

INTEGRATION COSTS

Integration costs are related to both the number of sources/subschemas and the number of entities to be integrated. We propose a cost model where three different types of costs are considered:

- Design costs (C^d), related to the production of the integrated schema and mappings to local sources. Design costs depend on the number of involved concepts in the set of schemas to be integrated.
- Execution time cost, including maintenance that can be further specialized into:
    - Fixed costs (C^f), that is costs related to the use of a wrapper mediator architecture; fixed costs represent the cost of the mediator;
    - Source costs (C^s) related to costs of software installed on sources included in the

integration, they represent the costs for the run time execution of the wrappers;

We do not consider the cost related to the evolution of schema due to the lack of consolidated results in both research and academy. With the above assumptions, the costs of integration are expressed by the following formula:

$$\mathrm{C} = C^f + C^s||S^c|| + (\sum_{m \in S_i \in S_j}(\left||m^{s_i}|\right| - 1) * C^d) \quad (9)$$

where

- S^c is the set of sources to be integrated;
- $||S^c||$ is the number of sources to be integrated;
- $\left||m^{s_i}|\right|$ is the number of schemas si in Sc including the entity m;
- C^d is the cost related to the design of the mediated schema.

Formula 9 assumes that the costs for schema maintenance related to new sources to be integrated increase linearly with the number of sources, and more than linearly with the number of concepts to be integrated. This is explained by considering that activities related to the design of the mediated schema and the mapping with local sources become more complex (and thus more costly) by definition of the integrated schema and mappings, when the same entity m is included in two or more schemas.

For what concerns the example of Figure 1, costs related to the integration of schemas S_1,S_3 and S_6 are equal to:

$$C_{1,3,6} = C^f + 3 \cdot C^s + 3C^d \quad (10)$$

since there are three sources and three entities (Department, City and Item) to be integrated. While integration costs related to schemas S_2, S_4 and S_6 are equal to

$$C_{2,4,6} = C^f + 3 \cdot C^s + 4C^d \quad (11)$$

Comparing equations 10 and 11, it results that $C_{1,3,6} < C_{2,4,6}$ due to the fact that in the last integration set the two entities Item and Type of Item are in common among all the involved schemas.

CHOICE OF THE OPTIMAL ARCHITECTURE(S)

In this section we show two approaches to select the best data integration architecture. The first approach assumes that a unique data integration solution is used, thus resulting in the choice of the optimal set of schemas to be included in the architecture. The second approach allows for a set of data integration solutions that can partially overlap. In both cases we assume the following inputs:

- N is the number of the data sources that can be integrated;
- $\hat{S}$, is the set of data sources;
- S_i is the schema of the i-th data source, there is one schema for each data source;
- C_{MAX} is the maximum cost for the whole integration effort;
- $IC_{S_{i_1 i_2 i_3 .. i_k}}$, is the potential information capacity (schema/instance based) evaluated by integrating $S_{i_1 i_2 i_3 .. i_k}$ schemas;
- $C_{S_{i_1 i_2 i_3 .. i_k}}$ is the integration cost related to the integration of $S_{i_1 i_2 i_3 .. i_k}$ different data sources.

It is worth noting that the algorithms we present are parametric w.r.t the two types of potential information capacity we defined, namely schema and instance based potential IC.

The optimal data integration architecture

The first algorithm identifies a unique best data integration solution. Our approach is based on the definition of a tree of acceptable solutions. The algorithm starts with a first target architecture, obtained by considering all the possible pairs of schemas $S_{i,j}$ so that $i < j$. For each solution, we evaluate the $PIC_{S_{i,j}}$. If $PIC_{S_{i,j}} = 0$, that is, if the integration of two schemas does not increase the overall information capacity, the solution is discarded and is no longer considered. The second step of the algorithm builds a new set of solutions starting from the previous ones. The generic solution is in the form of $S_{i,j,z}$ where $i < j < z$. For each solution we evaluate the ration of $PIC_{S_{i,j,z}}$ and $C_{S_{i,j,z}}$. If$\frac{PIC_{S_{i,j}}}{C_{S_{i,j}}} \geq \frac{PIC_{S_{i,j,z}}}{C_{S_{i,j,z}}}$ or $C_{S_{i,j,z}} > C_{MAX}$, the solution is discarded and no longer considered. The algorithm stops after $i = N$ iterations or when no new solution is generated. In both cases, starting from the leaves of this solution tree we select the data integration solution $\hat{S}$ so that $\forall \hat{S}^1 = \hat{S}$: $\frac{PIC_{\hat{S}}}{C_{\hat{S}}} \geq \frac{PIC_{\widehat{S1}}}{C_{\widehat{S1}}}$ and $C_{\hat{S}} \leq C_{MAX}$. In the worst case the total number of acceptable solutions is equal to$\sum_{k=1}^{N}(\frac{N!}{k!(N-k!)})$. In fact, at each step of the algorithm we produce a number of k combinations without repetitions and permutations of a set N. This value is equal to $\frac{N!}{k!(N-k)!}$. Consequently, the total number of possible solutions is the sum of all acceptable solutions generated in each step. Within the acceptable solutions, we focus mainly on the leaves of the solution tree we designed. In fact, the PIC is a monotonous non decreasing function, thus $PIC(\hat{S}) <= PIC(\hat{S}, S_j)\forall S_j$. Among leaves solutions we identify as the optimal data integration architecture the one with the highest PIC.

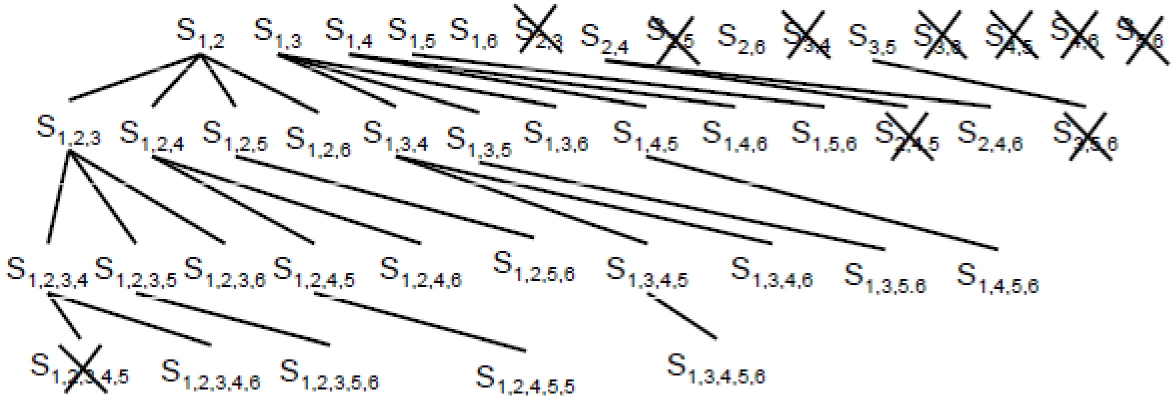


Figure 5 Example of optimal data integration architecture

In Figure 5 we apply the algorithm to the running case of Section *Running Case* where C_{MAX} = 150.000, C^f= 30, C^s = 2.000, Cd = 5, t = 4. It is worth noting that there are two different reasons to discard a solution. The first reason (e.g. solution $S_{2,3}$) is related to the absence of common entities among schemas (see Figure 2), the second one is related to the integration costs when they are higher than the maximum cost (e.g. solution $S_{1,2,3,4,5}$ whose cost is 171.081). According to the algorithm, we propose $S_{2,6}$ with a ratio $\frac{PIC^{sb}}{C}$= 0.377 as optimal solution.

Considering a set of data integration architectures

It is possible that the optimal solution $\hat{S}$ produces a cost $C_{\hat{S}} < C_{MAX}$. In this case we can change the perspective, no longer looking for the best unique solution, but a cluster of acceptable solutions so that$\sum\frac{PIC_{\widehat{S\prime}}}{C_{\widehat{S\prime}}}>\frac{PIC_{\hat{S}}}{C_{\hat{S}}}$ and $\sum C_{\widehat{S\prime}} \leq C_{MAX}$. To solve this problem we have to solve the following optimization problem:

$$\begin{array}{l} Min \sum X_i \\ \text{s.t.} \\ \sum X_i C_i < C^{MAX} \\ \sum X_i (\frac{PIC_i}{C_i}) > \frac{PIC_{\hat{S}}}{C_{\hat{S}}} \\ X_i \in \{0,1\} \end{array} \qquad (12)$$

The problem defined in formula (12) finds the minimum set of acceptable solutions whose sum of potential PICs is greater than the PIC of the unique solution. Equation 12 belongs to the family of coverage problems and its complexity is NP-complete. We apply this formula to the example of Section *Running Case*, by using a lp-solve[2], a mixed integer linear programming solver that solves pure linear, (mixed) integer/binary, semi-continuous and special ordered sets (SOS) models. Results define as optimal solution $\hat{S}$= ($S_{1,2,3,6}$, $S_{1,2,3,4,6}$). Unfortunately, this solution is not acceptable, since it is straightforward to show that the latter solution also includes the former one, that is $S_{1,2,3,6} \subset S_{1,2,3,4,6}$. Consequently, we introduce the concept of inclusion of a solution.

Definition 1. An acceptable solution Sis included by another acceptable solution S (S'≺ S) iff S_i ∈ S→ S_i ∈ S ∀ S_i ∈ S ∃S_z ∉ S' S_z ∈ S

By extending the formulation of the problem expressed by Formula 12, we include the following clause

$$\forall X_i, X_j : X_i \prec X_i X_i \;\; X_i \leq 1 \qquad (13)$$

In this way, X_i and X_i cannot be selected together. By applying the optimization problem to the set of acceptable solutions of Figure 5, the optimal set is $\hat{S}$ = $S_{1,3,5,6}$, $S_{1,2,3,4,6}$ with $\frac{PIC_{\hat{S}}}{C_{\hat{S}}}$ = 0.407 that is greater than 0.377 of the above mentioned solution.

RELATED WORKS

Data integration is a widely investigated research area, and significant literature is available (see e.g. [14, 13, 9]). However, to the best of our knowledge there are very few approaches that investigate the optimality of data integration architectures. [10] claims that the investment in schema management per new integrated sources and in heavy-weight middleware are reasons why user costs increase directly (linearly) with the user benefits, with the primary investment going to the middleware IT product and service providers. What is beneficial to end users, however, are integration technologies that truly demonstrate economies of scale, with costs of adding newer sources decreasing significantly as the total number of sources to be integrated increases, but no experiences or models are provided in order to support this intuition. In the last few years a few economic models have been proposed for the analysis of costs/benefits of other types of data architectures. For example, in [8] the economic contribution of tabular data sets to the design of data warehousing and database solutions is investigated. The framework proposed assumes that the business value contribution (conceptualized as *utility*) of data resources and the costs associated to managing them are influenced by the design characteristics of the data repositories and to processes used to create and manage them. Viewing the set of design characteristics as representing a design space, the authors assume that the economic performance can be maximized by determining the optimal point within this space, while considering applicable dependencies and constraints.

[2] http://sourceforge.net/projects/lpsolve/

CONCLUSION AND FUTURE WORKS

The data architecture of organizations is often broken up in a number of heterogeneous data sources; we can foresee that this trend will worsen in the future by the dynamic nature of current business activities and the evolution of networks and of the Web. In this context, it is quite difficult for organizations to exploit the whole information asset, due to the difficulty in obtaining a common view over data. To face this problem, the usage of data integration solutions is a mandatory step toward a more effective data architecture. While a lot of literature has been available for managing technical issues (e.g. schema matching [14]), to the best of our knowledge no analysis has been proposed on the economic sustainability in terms of cost/benefit evaluation of integration architectures based on data integration solutions. In this paper we propose three original results: i) a set of quality dimensions to evaluate a data architecture based on the concept of information capacity, ii) a cost function for evaluating the integration costs, and iii) two algorithms for identifying the optimal (set of) data architecture solution(s) maximizing the information capacity of the overall architecture within a given cost threshold. A simple but realistic example shows how our framework can be applied. Possible future work can be clustered in three research directions. The first one is related to the evaluation of more complex situations where the number of data sources to be integrated is high (e.g. all data sources of large organizations or national-level public administration). In this case, the number of acceptable solutions produced by our algorithms becomes intractable from a computational view-point. The second research direction is to consider in the cost analysis the use of design tools that simplify the creation of mediated schema [5] and consider the temporal dimension in the cost analysis (e.g. by considering maintenance costs and schema evolution events). The third area is related to the definition of a methodological framework for read-write access to the whole information asset.

REFERENCES

[1] A. Aboulnaga, P. Haas, M. Kandil, S. Lightstone, G. Lohman, V. Markl, I. Popivanov, and V. Raman. Automated statistics collection in db2 udb. In *VLDB ’04*, pages 1158–1169. VLDB Endowment, 2004.

[2] C. Batini, S. Ceri, and S. B. Navathe. *Conceptual Database Design: An Entity- Relationship Approach*. Benjamin/Cummings, 1992.

[3] C. Batini, C. Cappiello, C. Francalanci, A. Maurino: Methodologies for data quality assessment and improvement. ACM Comput. Surv. 41(3): (2009)

[4] C. Batini and M. Scannapieco. *Data Quality: Concepts, Methodologies and Techniques*. Data-Centric Systems and Applications. Springer, 2006.

[5] D. Beneventano, S. Bergamaschi, F. Guerra, and M. Vincini. The momis approach to information integration. In *ICEIS (1)*, pages 194–198, 2001.

[6] S. Bergamaschi and A. Maurino. Toward a unified view of data and services. In G. Vossen, D. D. E. Long, and J. X. Yu, editors, *WISE*, volume 5802 of *Lecture Notes in Computer Science*, pages 11–12. Springer, 2009.

[7] P. A. Bernstein and L. M. Haas. Information integration in the enterprise. *Commun. ACM*, 51(9):72–79, 2008.

[8] A. Even, G. Shankaranarayanan, and P. D. Berger. Economics-driven data management: An application to the design of tabular data sets. *IEEE Trans. Knowl. Data Eng.*, 19(6):818–831, 2007.

[9] A. Halevy, A. Rajaraman, and J. Ordille. Data integration: the teenage years. In *VLDB ’06*, pages 9–16. VLDB Endowment, 2006.

[10] A. Y. Halevy, N. Ashish, D. Bitton, M. Carey, D. Draper, J. Pollock, A. Rosenthal, and V. Sikka. Enterprise information integration: successes, challenges and controversies. In *SIGMOD ’05*, pages 778–787, New York, NY, USA, 2005. ACM.

[11] J. W. H.Garcia-Molina, J. D. Ullman. *Database Systems: The Complete Book (2nd Edition)*. Prentice Hall, 2008.

[12] W. H. Inmon. The data warehouse and data mining. *Commun. ACM*, 39(11):49– 50, 1996.

[13] M. Lenzerini. Data integration: a theoretical perspective. In *PODS ’02*, 233–246, USA, 2002. ACM.

[14] E. Rahm and P. A. Bernstein. A survey of approaches to automatic schema matching. *The VLDB Journal*, 10(4):334–350, 2001.

[15] G. Wiederhold. Mediators in the architecture of future information systems. *IEEE Computer*, 25(3):38–49, 1992.