# Optimal Tuple Merge is NP-Complete

Edward L. Robertson and Catharine M. Wyss

January 28, 2003

**Abstract**

The *Optimal Tuple Merge* (OTM) problem arises within the context of relational query language extensions to query and manipulate metadata as well as data. Such extensions include the ability to create *dynamic output schemas* from the input data. This flexibility is necessary for truly schema independent restructuring, however many null values may be introduced into the resultant data. Many of these "artifical" null values can be subsequently merged away. In this paper, we prove that the optimal merging case, in which the resulting relation contains as few tuples as possible, results in an NP-Complete problem. In spite of this, we characterize when an optimal (and unique) merge is easy to obtain, and identify at least one class of practically relevant relations where OTM is efficiently solvable.

**Keywords**: *databases*, *metadata*, *combinatorial problems*, *computational complexity*

## 1 Introduction

Recent frameworks for relational interoperability provide flexible methods for querying and restructuring both metadata and data in relational tables. This flexibility is necessary to satisfy the demands for *schema independent* queries that arise when combining semantically similar data from multiple, distinct sources. Applications of relational interoperability are widespread, and include support for Federated Information Systems (FIS), encompassing the ability to create and manage mediators, global schemas, and metadata dictionaries. Furthermore, recent extension to the relational query languages SQL and the RA enable real-time interoperability in a large federation of relational databases [3].

However, the added flexibility provided by frameworks for relational interoperability comes with a price. One required capability is the ability to promote a column of data to relational attributes (column headings). In order that this "transpose-like" operation on relations has a consistent semantics for any relational table, many null values may be introduced into the result. We would ideally like to merge these "artificial" null values as much as possible to obtain a relation with the minimum number of tuples.

As a motivating example, consider the relation in figure 1 (a). This relation tracks assignment grades for the students John, Jane, and Spot. After transposition, we obtain the relation in figure 1 (b). Such a transformation is important when translating from relational databases to spread sheets (or vice versa). In this case, we can merge as shown (figure 1 (c)).

| Name | Assignment | Grade |
|---|---|---|
| John | Asg1 | 78 |
| John | Asg2 | 73 |
| John | Asg3 | 88 |
| John | Asg4 | 82 |
| John | Asg5 | 84 |
| Jane | Asg1 | 50 |
| Jane | Asg2 | 53 |
| Jane | Asg3 | 48 |
| Jane | Asg5 | 67 |
| Spot | Asg1 | 90 |
| Spot | Asg2 | 85 |
| Spot | Asg3 | 96 |
| Spot | Asg4 | 89 |
| Spot | Asg5 | 98 |

(a)

Original relation

| Name | Asg1 | Asg2 | Asg3 | Asg4 | Asg5 |
|---|---|---|---|---|---|
| John | 78 | $\perp$ | $\perp$ | $\perp$ | $\perp$ |
| John | $\perp$ | 73 | $\perp$ | $\perp$ | $\perp$ |
| John | $\perp$ | $\perp$ | 88 | $\perp$ | $\perp$ |
| John | $\perp$ | $\perp$ | $\perp$ | 82 | $\perp$ |
| John | $\perp$ | $\perp$ | $\perp$ | $\perp$ | 84 |
| Jane | 50 | $\perp$ | $\perp$ | $\perp$ | $\perp$ |
| Jane | $\perp$ | 53 | $\perp$ | $\perp$ | $\perp$ |
| Jane | $\perp$ | $\perp$ | 48 | $\perp$ | $\perp$ |
| Jane | $\perp$ | $\perp$ | $\perp$ | $\perp$ | 67 |
| Spot | 90 | $\perp$ | $\perp$ | $\perp$ | $\perp$ |
| Spot | $\perp$ | 85 | $\perp$ | $\perp$ | $\perp$ |
| Spot | $\perp$ | $\perp$ | 96 | $\perp$ | $\perp$ |
| Spot | $\perp$ | $\perp$ | $\perp$ | 89 | $\perp$ |
| Spot | $\perp$ | $\perp$ | $\perp$ | $\perp$ | 98 |

(b)

Transposed relation (Asg1, Asg2, Asg3, Asg4, Asg5 data values have become column headings)

| Name | Asg1 | Asg2 | Asg3 | Asg4 | Asg5 |
|---|---|---|---|---|---|
| John | 78 | 73 | 88 | 82 | 84 |
| Jane | 50 | 53 | 48 | $\perp$ | 67 |
| Spot | 90 | 85 | 96 | 89 | 98 |

(c)

Optimal merged relation

Figure 1: Merge example.

What is special about this example is that there is exactly *one* assignment grade for each student. If the original relation had another tuple such as $\langle \mathtt{Name} : \mathtt{John}, \mathtt{Assignment} : \mathtt{Asg3}, \mathtt{Grade} : \mathtt{85} \rangle$, giving a conflicting grade for John's third assignment, the merge would no longer be unique. This intuition will be made precise in proposition 4.2. First, we will introduce some necessary terminology and consider the merging problem in more generality.

## 2 Terminology

The most basic elements within our formal framework are called *domain elements*. We assume a countably infinite domain of atomic elements, denoted ***dom***, which typically contains alphanumeric strings, such as `123`, `abc` or `SomeElement`. We will use teletype font to denote specific elements of ***dom***. In addition, we assume a special element, $\perp$, called the *null* element. We stipulate that $\perp \notin$ ***dom***.

**Definition 2.1**

A *tuple* is a mapping from ***dom*** to ***dom*** $\cup \{\perp\}$ that is $\perp$ at all but a *finite* number of domain elements.

A tuple may be represented explicitly using the notation $\langle A_1 : a_1, \ldots, A_n : a_n \rangle$ where $A_i \in$ ***dom*** and $a_i \in$ ***dom*** $\cup \{\perp\}$ $(1 \leq i \leq n)$. Alternatively (or in combination), we may use the notation $t[A]$ to mean $t(A)$, the $A$-th component of $t$. A canonical relation is a finite set of tuples. The active schema of a relation, $R$, denoted $asch(R)$, consists of domain elements, $A$, for which some tuple $t \in R$ assigns a non-null value to $A$.

**Definition 2.2 (see [2], definition 4.2)**

Let $R$ be a relation with $asch(r) = \{A_1, \ldots, A_n\}$.

1. Two tuples $t_1, t_2 \in R$ are *mergeable* provided for each $i = 1, \ldots, n$, either $t_1[A_i] = t_2[A_i]$ or one of $t_1[A_i]$ or $t_2[A_i]$ is a null value.

2. Suppose $t_1, t_2 \in R$ are mergeable. Then their *merge*, denoted $t = t_1 \odot t_2$, is given by

$$t[A_i] = \begin{cases} t_1[A_i] \text{ if } t_1[A_i] \neq \perp, \\ t_2[A_i] \text{ otherwise.} \end{cases} \qquad (1 \leq i \leq n)$$

In what follows, we consider the idea of an *optimal* merge result. A merge result is considered optimal if no smaller merge result can be obtained. This is made precise in definition 2.4.

**Definition 2.3**

Let $R, S$ be relations. Then we say $S$ is a *merge* of $R$ in case $S$ can be obtained from $R$ by a finite sequences of merges among mergeable tuples.

**Definition 2.4 (Optimal Merge)**

Let $R, S$ be relations. Then $S$ is *an optimal merge for* $R$ in case

1. $S$ is a merge of $R$, and
2. for all $S'$ such that $S'$ is a merge of $R$, we have that $|S| \leq |S'|$.

Note that, in general, there will be more than one optimal merge for a given relation $R$.

# 3 Optimal Tuple Merge is NP-Complete

Optimal merges for a relation are, in general, provably difficult to find. By this, we mean that the problem of finding an optimal merge counterpart for an arbitrary relation belongs to the well-known class of *NP-Complete* problems. Before proving this, we must precisely state the *Tuple Merge* and *Optimal Tuple Merge* problems more formally.

**Definition 3.1**

1. The *Tuple Merge Decision* (TMD) problem is thus: given a relation $R$ and a natural number $l$, is there a merge of $R$ of size $l$ (i.e. an $l$-merge)?
2. The *Optimal Tuple Merge* (OTM) problem is to find an optimal merge of a given relation $R$.

If we can determine whether a relation has a merge of size $l$ in time $T(l)$, we can find such a merge in time $O(|R|^3 \cdot T(l) \cdot l)$. Thus, a polynomial-time solution to to TMD gives a polynomial-time solution to OTM, since we need only determine the least number in $\{1, \ldots, |R|\}$ for which a merge exists, and then compute such a merge.

**Theorem 3.1 (Tuple Merge Decision is NP-Complete)**

Let $R$ be a relation and $l$ a natural number. Then the problem of determining whether $R$ has an $l$-merge is NP-complete.

**Proof of Theorem 3.1**:

We will give a class of $\langle$relation, $l\rangle$ pairs for which the problem of deciding whether there is an

$l$-merge is equivalent to finding a solution to the known NP-Complete decision problem *Hitting Set* ([1], page 222).

Hitting Set involves a finite set $S = \{1, \ldots, m\}$ and a collection of sets $C_1, \ldots, C_n \subseteq S$. In addition, we are given an integer $k \leq m$. The problem asks: is there a set $S' \subseteq S$ where $|S'| = k$ and for all $j, 1 \leq j \leq n$, $C_j \cap S' \neq \emptyset$?

Given an instance of Hitting Set, we define the following relation, $R$. The active schema of $R$ will be $\{N, B, C_1, \ldots, C_n\}$.[1] In addition $R$ contains the following tuples:

1. $m$ tuples $E_i := \langle N : i, B : \perp, C_1 : x_1, \ldots, C_n : x_n \rangle$ for $1 \leq i \leq m$, where for $1 \leq j \leq n$

$$x_j = \begin{cases} 1 \text{ in case } j \in C_j, \text{ and} \\ 0 \text{ otherwise;} \end{cases}$$

2. $n$ tuples $C_q := \langle N : \perp, B : 0, C_1 : \perp, \ldots, C_q : 1, \ldots, C_n : \perp \rangle$ for $1 \leq q \leq n$; and

3. $m - k$ tuples $B_p := \langle N : \perp, B : p, C_1 : \perp, \ldots, C_n : \perp \rangle$ for $1 \leq p \leq m - k$.

Given $R$, let $l = m$. We claim that there is an $l$-merge of $R$ exactly in case there is a hitting set for the $C_i$ of size $k$.

Any merge of $R$ will be of size $\geq m$ since none of the $E_i$ tuples can be merged with each other (they all differ on the $N$ component).

Additionally, because the $B$ component of the $B_p$ tuples is distinct from the $B$ component of the $C_q$ tuples, no $B_p$ tuple can merge with any $C_q$ tuple. However, the $B_p$ tuples can merge with the $E_i$ tuples. Assume this is done, leaving $m - (m - k) = k$ $E_i$ tuples unmerged with $B_p$'s that need to be merged with the $C_q$ tuples.

It should be clear that, by definition, a $C_q$ can only merge with an $E_i$ in case $i \in C_q$; hence, we can see that *all* remaining $E_i$ tuples will be merged exactly in case there is at least one Hitting Set of size $k$ for the $C_q$ sets (this Hitting Set will in fact be given as the set of $N$ components of the $E_i$ tuples that merged with the $C_q$ tuples).

[1] We will overload the names "$C_1$" through "$C_n$" to denote both the subsets of $S$ as well as the corresponding column attributes of $R$ so that the following exposition is clearer.

HS case 1:
$S = \{1, 2, 3, 4\}$
$C_1 = \{1, 2, 4\}$
$C_2 = \{2, 3, 4\}$
$k = 1$

R

| | N | B | $C_1$ | $C_2$ |
|---|---|---|---|---|
| $E_1$ | 1 | $\perp$ | 1 | 0 |
| $E_2$ | 2 | $\perp$ | 1 | 1 |
| $E_3$ | 3 | $\perp$ | 0 | 1 |
| $E_4$ | 4 | $\perp$ | 1 | 1 |
| $C_1$ | $\perp$ | 0 | 1 | $\perp$ |
| $C_2$ | $\perp$ | 0 | $\perp$ | 1 |
| $B_1$ | $\perp$ | 1 | $\perp$ | $\perp$ |
| $B_2$ | $\perp$ | 2 | $\perp$ | $\perp$ |
| $B_3$ | $\perp$ | 3 | $\perp$ | $\perp$ |

A 4−merge of R:

| | N | B | $C_1$ | $C_2$ |
|---|---|---|---|---|
| $B_1 \odot E_1$ | 1 | 1 | 1 | 0 |
| $B_2 \odot E_2$ | 2 | 2 | 1 | 1 |
| $B_3 \odot E_3$ | 3 | 3 | 0 | 1 |
| $C_1 \odot C_2 \odot E_4$ | $\boxed{4}$ | 0 | 1 | 1 |

(Hitting set highlighted.)

(a)

HS case 2:
$S = \{1, 2, 3, 4\}$
$C_1 = \{1, 3\}$
$C_2 = \{2, 4\}$
$k = 1$

R

| | N | B | $C_1$ | $C_2$ |
|---|---|---|---|---|
| $E_1$ | 1 | $\perp$ | 1 | 0 |
| $E_2$ | 2 | $\perp$ | 0 | 1 |
| $E_3$ | 3 | $\perp$ | 1 | 0 |
| $E_4$ | 4 | $\perp$ | 0 | 1 |
| $C_1$ | $\perp$ | 0 | 1 | $\perp$ |
| $C_2$ | $\perp$ | 0 | $\perp$ | 1 |
| $B_1$ | $\perp$ | 1 | $\perp$ | $\perp$ |
| $B_2$ | $\perp$ | 2 | $\perp$ | $\perp$ |
| $B_3$ | $\perp$ | 3 | $\perp$ | $\perp$ |

an optimal merge of R:

| | N | B | $C_1$ | $C_2$ |
|---|---|---|---|---|
| $B_1 \odot E_1$ | 1 | 1 | 1 | 0 |
| $B_2 \odot E_2$ | 2 | 2 | 0 | 1 |
| $B_3 \odot E_3$ | 3 | 3 | 1 | 0 |
| $C_1 \odot E_4$ | 4 | 0 | 0 | 1 |
| $C_2$ | $\perp$ | 0 | 1 | $\perp$ |

No 4−merge of R exists.

(b)

Figure 2: Hitting Set translations to Tuple Merge Decision.

This shows that an $l$-merge of $R$ determines a hitting set of size $k$. Conversely, a hitting set of size $k$ will clearly determine an $l$-merge of the relation $R$ as defined above. Hence, Tuple Merge is *at least* as difficult as Hitting Set.

On the other hand, Tuple Merge is in NP. The easiest way to see this involves the idea of a *merge-mapping*. Given relation $R$, and natural number $l$, an *$l$-merge-mapping* for $R$ is a mapping from the tuples of $R$ into $\{1, \ldots, l\}$ that describes an $l$-merge of $R$ (tuples mapped to the same number are to be merged). Suppose we have a relation $R$ and a natural number $l$, and we guess an $l$-merge-mapping for $R$. In this case, we can verify whether the mapping indeed describes a well-defined $l$-merge of $R$ in time $O(|R| \cdot l)$ by attempting to build the $l$-merge. Thus, TMD is solvable in NP.

**Corollary 3.1**

OTM is NP-complete.

**Example 3.1**

Figure 2 (a) and (b) give two Hitting Set problems and corresponding relations $R$. Figure 2 (a) gives an example where an $l$-merge exists; figure 2 (b) gives an example where an $l$-merge does not exist. In figure 2 (a), the elements of the hitting set are highlighted in the $l$-merge relation.

# 4 Further Results

Although the above result would appear to be seriously limiting in terms of the viability of the merge operation for federated relations, there is a broad class of relations which admit a unique (and easily identifiable) optimal merge. Furthermore, the description of this class gives us a polynomial-time test to determine whether the merge of a relation is unique.

**Proposition 4.1**

Let $R$ be a relation. Then $R$ has a unique non-trivial optimal merge exactly in case the following condition holds: for every triple $t_1, t_2, t_3$ of tuples in $R$, if at least two pairs involving $t_1, t_2, t_3$ are mergeable, then all of $t_1, t_2$ and $t_3$ are mergeable.

**Proof of Proposition 4.1**:

Both directions are reasonably straightforward using proof by contradiction. We will show the condition is sufficient to guarantee a unique merge. Indeed, suppose not: *i.e.* the condition holds but there are at least two distinct non-trivial optimal merges of $R$, namely $S$ and $S'$. Since $S$ and $S'$ are both non-trivial and optimal yet distinct, there are tuples $t, x_1, \dots, x_n, y_1, \dots, y_m \in R$ such that $t_S = t \odot x_1 \odot \cdots \odot x_n \in S$ and $t_{S'} = t \odot y_1 \odot \cdots \odot y_m \in S'$.

> *Case 1.* If $t_S$ and $t_{S'}$ are not mergeable, then there must be $x_i$ and $y_j$ which are not mergeable, and $t$, $x_i$ and $y_j$ break the condition.

> *Case 2.* If $t_S$ and $t_{S'}$ are mergeable, then consider $S$ alone, where $t \odot x_1 \odot \cdots \odot x_n \odot y_1 \cdots \odot y_m$ could replace $t \odot x_1 \odot \cdots \odot x_n$ with the $y_j$'s being removed from other tuples where they occur. Since $S$ is minimal, this cannot empty out any other tuples, so there is some $y_J$ and $z_1, \dots, z_k$ where $y_J \odot z_1 \odot \cdots \odot z_k$ is in the original $S$ but is not mergeable with $t \odot x_1 \odot \cdots \odot x_n$. In this case, $y_J$, some $z_k$, and either $t$ or some $x_i$ comprise a triple that violates the condition.

**Corollary 4.1**

Given $R$, a relation such that $|R| = n$, we can test whether there is a unique optimal merge for $R$ using at most $O(n^3)$ tuple comparisons.

The class of relations having a unique optimal merge includes a sub-class particularly appropriate for use with relational transposition.[2] The description of this class requires recourse to the notion of a *superkey* of a relation: a set of attributes whose values uniquely identify every tuple in the relation. Definitions 4.1, 4.2 and proposition 4.2 (below) gives a precise formal account of why the merging in figure 1 is well-defined.

**Definition 4.1**

Let $R(A, B, C_1, \ldots, C_n)$ be a relation schema. Then the *transpose of $A$ on $B$ of $R$*, denoted $\tau_A^B(R)$ is given as follows.

1. The active schema of $\tau_A^B(R)$ is $\{v : v \in \Pi_B(R), v \neq \perp\} \cup asch(R)$,[3] and
2. whenever there is $t = \langle A : a, B : b, C_1 : c_1, \ldots, C_n : c_n\rangle \in R$ where $b \neq \perp$ then the tuple $\langle A : a, B : b, C_1 : c_1, \ldots, C_n : c_n, b : a\rangle \in \tau_A^B(R)$.

**Definition 4.2**

Let $R(A_1, \ldots, A_n)$ be a relational schema. Then $\amalg_{A_i}(R)$, the *drop projection of $R$ on $A_i$*, is defined as $\amalg_{A_i}(R) = \Pi_{asch(R)-\{A_i\}}(R)$.

**Proposition 4.2**

Let $R$ be a relation with $asch(R) = \{A, B, C_1, \ldots, C_n\}$. Let $R' := \amalg_{A,B}(\tau_A^B(R))$. Suppose furthermore that $\{B, C_1, \ldots, C_n\}$ is a *superkey* for $R$. Then $R'$ has exactly one non-trivial optimal merge.

**Proof of Proposition 4.2**:

Suppose $R$, $R'$ are as above and the condition holds. Since the $\{B, C_1, \ldots, C_n\}$ components uniquely identify tuples in $R$, there will be at most *one* $A$ value in $R$ for every group of distinct $B, C_1, \ldots, C_n$ values. Consider $R'$. Let $\Pi_B(R) = \{\langle b_1\rangle, \ldots, \langle b_m\rangle\}$. Then the active schema of $R'$ is $\{b_1, \ldots, b_m\} \cup \{C_1, \ldots, C_m\}$ (note we dropped the $A$ and $B$ columns in $R'$). We can partition $R'$ into sub-relations

[2] Every relation can be seen as the transpose of *some* relation, thus the following result is general.

[3] $\Pi$ denotes ordinary relational projection.

$R'_1, \ldots, R'_k$ based on distinct $C_1, \ldots, C_n$ values. For each of these partitions $R'_i$ ($1 \le i \le k$), at most one tuple has a non-null $b_j$ component for all $j$, $1 \le j \le m$. This means each of these partitions can be fully merged in exactly one way, resulting in one tuple for each $R'_i$. Altogether we will have $k$ tuples in the optimal merge of $R'$ as a result.

# 5 Conclusions

Given propositions 4.1 and 4.2, we can easily determine whether merging the result of a transpose is well-defined. In fact, according to proposition 4.2, the best case involves checking for a superkey (which is linear in the size of the relation).

We can incorporate this into our relational interoperability framework as follows. We provide a fast merge operator for relations, $\odot$, that checks the condition in proposition 4.2. In case the condition holds, the unique optimally merged result is returned. Otherwise, we can return some heuristically determined but not necessarily optimal merge (or simply issue an error). In this way, we can meet the user's expectations for the transposition of a relation in a broad range of cases which will suffice in practice since "artificial" null values will most often arise during relational interoperation.

# References


[1] M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman and Company, 1979.

[2] L. V. S. Lakshmanan, F. Sadri and S. N. Subramanian. *SchemaSQL – An Extension to SQL for Multidatabase Interoperability.* ACM Transactions on Database Systems 26:4, pp. 476-519. December, 2001.

[3] C. M. Wyss. *Relational Interoperability*. PhD Thesis, Indiana University at Bloomington, August 2002. Available at `http://www.fisql.com/`.