Uncertain Schema Matching Based on Interval Fuzzy Similarities

WENG Nian-feng, DIAO Xing-chun

Abstract

Schema matching is very important in many database applications. Due to the fuzziness of knowledge representation, schema matching contains uncertainty inherently. In order to manage the uncertainty of schema matching process, fuzzy decision making theory is employed and the similarity of schema element pair is given in the form of interval fuzzy number. According to the concept of composite schema matching, interval fuzzy similarities are constructed and a candidate mapping selection procedure is proposed based on the priority of interval fuzzy similarities. After candidate mappings are selected, we propose a semantic conflict elimination procedure to remove false positive candidate mappings. Many existing schema matching algorithms can only identify 1:1 mappings, while our algorithm can identify both 1:1 and 1:n mappings. The quality of our uncertain schema matching algorithm is verified by experiments.

Keywords: Schema Matching, Uncertainty, Fuzzy Decision Making, Interval Fuzzy Similarity

1. Introduction

Schema matching is very important in many database applications, such as data integration, e-business, data warehousing and semantic query processing [1, 2, 3, 4]. Given two data schemas, schema matching process is to identify schema element correspondences according to different heuristics. Heuristics we can employ include the name of schema elements, structure information of schemas, application semantics and so on [5, 6]. For any single kind of heuristics is not enough for identifying schema element correspondences accurately, most schema matching algorithms use as much heuristics as possible which leads to the classification of hybrid matcher and composite matcher in [1]. In hybrid matcher [7, 8], multiple matching criteria are integrated directly. While in composite matcher [9, 10], results of independently executed matchers are combined. Although many schema matchers are available, to obtain accurate schema mappings is still very hard because of the inherent uncertainty of schema matching process [11]. Based on the concept of composite matcher, we employ fuzzy decision making theory to model schema matching process and propose an algorithm named Interval Fuzzy Ranking (IFR for short) to manage the uncertainty of schema matching. Experiments have shown that our proposed algorithm can generate high quality matching results.

The rest of this paper is organized as follows. In section 2, we analyze the uncertainty in schema matching process. In section 3, we propose a framework of uncertain schema matching based on fuzzy decision making theory. We give the detailed algorithms of uncertain schema matching in section 4. In section 5, we use open data schemas to verify our algorithms. At last, we conclude in section 6.

2. Uncertainty in schema matching process

Data schema consists of labels and structure and its syntax cannot represent semantics precisely which makes schema matching a hard work. At the beginning, schema matching was processed by domain experts and schema designers. But as the scale of data schema grows, it is hard for any expert to identify schema correspondences rapidly and correctly. As a result, computer aided schema matching was proposed. During the computer aided schema matching process, programs filter the candidate mappings which are less similar out and experts intervene in the matching process to provide more heuristics or verify the final matching results identified by machines. As web data management, e-business and cloud computing arise, there are a large amount of data sources online to be managed. Once data schema is defined, it is unlikely to be modified. Otherwise it will lead to a large amount of
data transformation. Meanwhile the personnel variety is always pervasive. So we cannot insure that experts or schema designers are always at hand. As a result, automatic schema matching is proposed. It is hoped that automatic schema matchers try their best to identify schema mappings without human intervention. But due to the semantic heterogeneity of data schemas, schema matching process contains uncertainty inherently. Some schema matcher employs the name of schema elements as heuristics. But the occurrence of synonyms and homonyms will result in erroneous mappings. For example, it will probably identify Pets.age and Kids.age as a candidate mapping according to string similarity. And it will probably fail to identify Employee.salary and Personnel.pay as a candidate mapping. While matchers use other heuristics such as dictionaries and thesauri will generate better matching results [12]. In order to improve the quality of schema matching result, more kinds of heuristics are used. It is certain that different kinds of heuristics will yield different matching results. But it is uncertain which kind of heuristics will yield best matching results.

3. The framework of uncertain schema matching

In the composite schema matching process, different matchers execute independently, each generates a similarity matrix as its matching result. Then the composite matcher combines results generated by these matchers and selects candidate mappings as the final matching result.

**Definition 1:** Given a schema matcher $A$ and two schemas $S$ and $T$ containing $m$ and $n$ attributes respectively, for each $s_i \in S$, $0 \leq i < m$, and each $t_j \in T$, $0 \leq j < n$, if $\text{Sim}_A(s_i, t_j) \geq 0$, where $\text{Sim}_A(s_i, t_j)$ is the similarity of schema element pair $(s_i, t_j)$ given by $A$, then $(s_i, t_j)$ is a candidate mapping.

**Definition 2:** Given a schema matcher $A$ and two schemas $S$ and $T$ containing $m$ and $n$ attributes respectively, the match result of schema matcher $A$ is a similarity matrix $M_{n \times m}$, where $m_{ij} = \text{Sim}_A(s_i, t_j)$, $0 \leq i < m, 0 \leq j < n$.

**Definition 3:** Given a set $\mathcal{A}$ of schema matchers of cardinality $l$ and two schemas $S$ and $T$ with $m$ and $n$ attributes respectively, composite schema matcher execute each schema matcher $A_i \in \mathcal{A}$, $0 \leq k < l$, independently, each return a similarity matrix as its result. Then assembling these results to a similarity cub $\text{Sim}_C_{m \times n \times l}$, where $c_{ijkl} = \text{Sim}_A(s_i, t_j)$, $0 \leq i < m, 0 \leq j < n, 0 \leq k < l$, is the similarity of candidate mapping $(s_i, t_j)$ given by schema matcher $A_k$.

Given several matching results, the combination process is to decide which candidate mappings to choose according to their similarities. So the combination process is also a decision making process. We use fuzzy decision making theory to model schema matching process. In a decision making process, expert engages in a pair wise comparison of candidates. The pair wise comparison generates a matrix called judgment matrix [14]. The decision making process is to decide the priority of candidates according to the preferences given by expert.

**Definition 4:** Given a judgment matrix $P=(p_{ij})_{n \times m}$, if $p_{ij} \cdot p_{ij} = 1$, $p_{ij} \geq 0$, then $P$ is a complementary judgment matrix.

If a judgment matrix $P_{n \times m}$ is a complementary judgment matrix, where $p_{ij}$ is the preference value, then we can compute priorities of candidates according to [13], i.e. given a complementary matrix $P=(p_{ij})_{n \times m}$, we can compute a priority vector $V=(v_1, v_2, \ldots, v_n)^T$, where

$$v_i = \frac{1}{n(n-1)} \left( \sum_{j=1}^{n} p_{ji} + \frac{n-1}{2} \right), 0 \leq i < n.$$  \hspace{1cm} (1)

If $v_i \geq v_j$ according to the priority vector, then the $i$-th candidate is prior to the $j$-th candidate.

In order to model the uncertainty in schema matching process, similarities of each schema correspondence given by independent matchers are aggregated to an interval fuzzy number. We formalize it as follows.

**Definition 5:** If $a=[a^-, a^+] = \{ x \mid a^- \leq x \leq a^+ \}$, then $a$ is an interval fuzzy number. Specially, if $a^- = a^+$, then $a$ degenerates to a real number.

**Definition 6** [14]: Given $a=[a^-, a^+]$, $b=[b^-, b^+]$ are interval fuzzy numbers, let $l_a=a^+-a^-$, $l_b=b^+-b^-$,

$$p(a \geq b) = \frac{\min\{l_a, l_b\} + \max\{a^- - b^-, 0\}}{l_a + l_b}$$  \hspace{1cm} (2)
is the possibility of \(a \geq b\).

\[
p(b \geq a) = \frac{\min\{l_1, l_4, \max(b^- - a^-, 0)\}}{l_1 + l_4}
\]

(3)

is the possibility of \(b \geq a\).

**Definition 7:** Given \(n\) interval fuzzy numbers \(\tau_k, 0 \leq k < n\), performing a pair wise comparison on these interval fuzzy numbers. The comparison process generates a matrix \(P_{n \times n}\), where \(p_{ij}\) is the possibility of \(\tau_i \geq \tau_j\), called possibility matrix.

**Theorem 1:** Possibility matrix is a complementary matrix.

**Proof.** We need to proof \(p(a \geq b) + p(b \geq a) = 1\). According to definition 5, \(l_k = a_k - a\geq 0\), \(l_k = b_k - b \geq 0\), so \(l_k + l_k \geq 0\). It is obvious that \(\max(a^- - b^-), 0\) \(\geq 0\), then \(\min(1, 1, \max(a^- - b^-), 0) \geq 0\). So we can conclude that \(p(a \geq b) \geq 0\). In the same way, we get that \(p(b \geq a) \geq 0\).

Then we proof \(p(a \geq b) + p(b \geq a) = 1\). According to formula (2) and formula (3),

\[
p(a \geq b) + p(b \geq a) = \frac{\min\{l_1, l_4, \max(a^- - b^-), 0\}}{l_1 + l_4} + \frac{\min\{l_1, l_4, \max(b^- - a^-), 0\}}{l_1 + l_4}
\]

(4)

(a) Let \(a^- - b^- < 0\), that is \(a^- < b^-\). According to definition 5, we get that \(b^+ > a^-\), that is \(a^+ - a^- \geq 0\). For \(b^- - a^+ < (a^+ - a^-) + (a^- - b^-) = a^+ - a^- + b^- - b^+ > 0\), then \(b^- - a^- + b^- - b^+ > 1\). Then we get that

\[
p(a \geq b) + p(b \geq a) = \min\{1, 0\} + \min\{1, \frac{b^- - a^-}{a^- - a^+ + b^- - b^+}\} = 1.
\]

(b) Let \(b^- - a^- < 0\), that is \(b^- < a^-\). Then \(a^- - b^+ > 0\). For \(a^- - b^- > (a^- - b^-) + (b^- - a^-) = a^- - a^- + b^- - b^+ > 0\), then \(a^- - a^- + b^- - b^+ > 1\). Then we get that

\[
p(a \geq b) + p(b \geq a) = \min\{1, 1\} = 1.
\]

(c) Let \(a^- - b^- > 0\) and \(b^- - a^- > 0\), then \(a^- - b^- < (a^- - b^-) + (b^- - a^-) = a^- - a^- + b^- - b^+ > 0\), and \(b^- - a^- < (b^- - a^-) + (a^- - b^-) = a^- - a^- + b^- - b^+ > 0\).

Then we get \(0 < \frac{b^- - a^-}{a^- - a^- + b^- - b^+} = \frac{b^- - a^-}{a^- - a^- + b^- - b^+} < 1\). So

\[
p(a \geq b) + p(b \geq a) = \min\{\frac{a^- - b^-}{a^- - a^- + b^- - b^+}, 1\} + \min\{\frac{b^- - a^-}{a^- - a^- + b^- - b^+}, 1\}
\]

\[
= \frac{a^- - b^-}{a^- - a^- + b^- - b^+} + \frac{b^- - a^-}{a^- - a^- + b^- - b^+} = 1.
\]

According to (a), (b) and (c), we can conclude that \(p(a \geq b) + p(b \geq a) = 1\).

4. Algorithm of uncertain schema matching

According to the concept of composite matcher, we take advantage of different schema matching algorithms and use decision making theory to manage uncertainty in schema matching process. Our uncertain schema matching algorithm consists of three steps. In the first step, interval fuzzy similarities are constructed from a set of real similarities given by different schema matchers. In the second step, candidate mappings are selected according to their interval fuzzy similarities. In the third step, semantic conflict elimination process is executed to remove candidate mappings which conflict with any candidate mapping.
4.1. Constructing interval fuzzy similarities

First of all, we need to construct interval fuzzy similarities from similarity matrixes generated by independent matchers. Assume that similarities generated by these matchers are all between 0 and 1. Otherwise we need an extra normalize step. Given a similarity cub generated by several schema matchers, for each candidate mapping we compute the mean of similarities generated by different matchers as the threshold. Then we compute the mean of similarities which are less than the threshold as the lower of this interval fuzzy number, say $a^-$. Finally, we compute the mean of similarities which are greater than the threshold as the upper of this interval fuzzy number, say $a^+$. Then a interval fuzzy similarity $a=[a^-, a^+]$ is constructed. We compute interval fuzzy similarity for each candidate mapping. Then we get an interval fuzzy similarity matrix. The detailed procedure is illustrated in Table 1.

<table>
<thead>
<tr>
<th>Table 1. Interval fuzzy similarity construction procedure</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Interval Fuzzy Similarity Construction</strong></td>
</tr>
<tr>
<td>input: similarity cub $\text{simCub}[[m]][[n]][[l]]$</td>
</tr>
<tr>
<td>output: triangular fuzzy similarity matrix $\text{fuzzySimMatrix}[[m]][[n]]$</td>
</tr>
<tr>
<td>1: for each $(s_i, t_j) \in S \times T$, $0 \leq i &lt; m$, $0 \leq j &lt; n$</td>
</tr>
<tr>
<td>2: $\text{threshold} \leftarrow \frac{1}{mn} \sum_{i,j} \text{simCub}[[i]][[j]][[k]]$;</td>
</tr>
<tr>
<td>3: $\text{fuzzySimMatrix}[[i]][[j]].\text{upper} \leftarrow$ the mean of similarities of $(s_i, t_j)$ which are greater than threshold;</td>
</tr>
<tr>
<td>4: $\text{fuzzySimMatrix}[[i]][[j]].\text{lower} \leftarrow$ the mean of similarities of $(s_i, t_j)$ which are less than threshold;</td>
</tr>
<tr>
<td>5: return $\text{fuzzySimMatrix}$;</td>
</tr>
</tbody>
</table>

4.2. Selecting candidate mappings

After interval fuzzy similarity matrix is constructed, we use formula (2) to compute the judgment matrix and use formula (1) to rank candidate mappings according to theorem 1. After ranking operation is finished, we need to select candidate mappings according to ranking results. Most schema matchers choose candidate mappings with similarities exceed a threshold [8, 9] as their matching results. The drawback of this strategy is that it is difficult to choose a proper threshold. If this threshold is too small, then too many false positive candidate mappings are selected. If this threshold is too large, then true positive candidate mappings are filtered out. Some schema matchers rank candidate mappings by their similarities and choose top-k candidate mappings as the final matching result [15]. Be the same with the threshold strategy, it is difficult to choose a proper $k$ value. In our algorithm, we rank candidate mappings by their interval fuzzy similarities and use a mechanism to choose candidate mappings in light of their relative priority but not a threshold or $k$ value. And our algorithm can identify both 1:1 and 1:n mappings inherently.

The ranking operation leads to two strategies. The first strategy ranks each row of similarity matrix, i.e. for each $s_i \in S$, ranking candidate mappings $(s_i, t_j)$, $t_j \in T$. The second strategy ranks each column of similarity matrix, i.e. for each $t_j \in T$, ranking candidate mappings $(s_i, t_j)$, $s_i \in S$. We call them by row strategy and by column strategy respectively. First, in the by row strategy we select the first target schema element which precedes others for each source schema element. Then we count each target schema element’s presence. If one target schema element presents $k$ times, there are $k$ source schema elements corresponding to the target schema element. Checking whether each of these $k$ source schema elements is in the top-$k$ list corresponding to the target schema element in the by column strategy. If so, a candidate mapping consists of this source schema element and target schema element is added into the derived schema mapping list. Second, we execute above procedure in the opposite direction, i.e. in the by column strategy we select the first source schema element which precedes others for each target schema element. Then we count each source schema element’s presence. If one source schema element presents $k$ times, there are $k$ target schema elements corresponding to the source schema element. Checking whether each of these $k$ target schema element is in the top-$k$ list corresponding to the source schema element in the by row strategy. If so, a candidate mapping consists of this source schema element and target schema element is added into the derived schema mapping list. The detailed procedure is illustrated in Table 2.


Table 2. Candidate mapping selection procedure

<table>
<thead>
<tr>
<th>Candidate Mapping Selection</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>input:</strong> an interval fuzzy similarity matrix $\text{fuzzySimMatrix}[m][n]$</td>
</tr>
<tr>
<td><strong>output:</strong> a list of derived mappings $\text{derivedMappings}$</td>
</tr>
</tbody>
</table>

1: ranking each row of $\text{fuzzySimMatrix}$;  
2: ranking each column of $\text{fuzzySimMatrix}$;  
3: for each source element  
4: get the target element which is ranked first in the by row strategy;  
5: for each distinct target element $t$ in selected target element list  
6: $k \leftarrow$ times of presence of $t$;  
7: for each source element $s$ corresponding to $t$  
8: $h \leftarrow$ position of $s$ in the ranked list corresponding to $t$ in by column strategy;  
9: if $h \leq k$ and $(s, t)$ is not in $\text{derivedMappings}$ then  
10: add $(s, t)$ into $\text{derivedMappings}$;  
11: for each target element  
12: get the source element which is ranked first in the by column strategy;  
13: for each distinct source element $s$ in the selected source element list  
14: $k \leftarrow$ the times of presence of $s$;  
15: for each target element $t$ correspond to $s$  
16: $h \leftarrow$ position of $t$ in the ranked list corresponding to $s$ in by row strategy;  
17: if $h \leq k$ and $(s, t)$ is not in $\text{derivedMappings}$ then  
18: add $(s, t)$ into $\text{derivedMappings}$;  
19: return $\text{derivedMappings}$;

In our algorithm, 1:1 mapping is a special case of 1:n mapping. The candidate mapping selection procedure can identify both 1:1 and 1:n mappings effectively, i.e. if more than one source schema elements are mapped to the same target schema element in the by row strategy and these source schema elements are ranked high in the by column strategy corresponding to the target schema element, then we can confirm the existence of 1:n mappings. In the same way, if many target schema elements are mapped to the same source schema element in the by column strategy and these target schema elements are ranked high in the by row strategy corresponding to the source schema element, then we can confirm the existence of 1:n mappings. The effectiveness of IFR algorithm in identifying 1:n mappings is verified by experiments in section 5.

4.3. Eliminating semantic conflicts

After the candidate mapping selection procedure, a list of candidate mappings is derived. In derived candidate mappings, there are semantic conflicts which need to be eliminated. We deal with two kinds of semantic conflicts. One kind of semantic conflict is that one schema element is mapped to several sibling schema elements. The other kind of semantic conflict is that one schema element is mapped to schema element and its children at the same time. We can construct a relation from one schema element and its children. Then these two kinds of semantic conflict violate the first norm form, i.e. a relation does not contain duplicate columns. In order to deal with these two kinds of semantic conflicts, we only reserve one candidate mapping from each semantic conflict which precedes others in the ranking results and just remove other candidate mappings from the derived candidate mapping list. Specifically, if one source schema element is mapped to sibling schema elements or parent and children schema elements in the target schema at the same time in by column strategy, then IFR reserves candidate mapping containing target schema element which is ranked high in the by row strategy corresponding to the source schema element and removes other candidate mappings from the derived mapping list. If one target schema element is mapped to sibling schema elements or parent and children schema elements in source schema at the same time in by row strategy, then IFR reserves candidate mapping containing source schema element which is ranked high in the by column strategy corresponding to the target schema element and removes other candidate mappings from the derived mapping list. The detailed procedure is illustrated in Table 3.
Table 3. Semantic conflict elimination procedure

<table>
<thead>
<tr>
<th>Semantic Conflict Elimination</th>
</tr>
</thead>
<tbody>
<tr>
<td>input: a list of derived mappings \textit{derivedMappings}</td>
</tr>
<tr>
<td>output: a list of derived mappings \textit{derivedMappings}</td>
</tr>
<tr>
<td>1: \textit{size} ← the size of \textit{derivedMappings};</td>
</tr>
<tr>
<td>2: for \textit{i=0} to \textit{size}</td>
</tr>
<tr>
<td>3: \textit{(s,t)} ← get the \textit{i}-th candidate mapping in \textit{derivedMappings};</td>
</tr>
<tr>
<td>4: for \textit{j=i} to \textit{size}</td>
</tr>
<tr>
<td>5: \textit{(s’,t’)} ← get the \textit{j}-th candidate mapping in \textit{derivedMappings};</td>
</tr>
<tr>
<td>6: if \textit{t} is child of \textit{t’} or \textit{t’} is child of \textit{t} or \textit{t} is sibling of \textit{t’}</td>
</tr>
<tr>
<td>7: if \textit{t} is ranked ahead of \textit{t’} corresponding to \textit{s} in by row strategy then</td>
</tr>
<tr>
<td>8: remove \textit{(s’,t’)} from \textit{derivedMappings};</td>
</tr>
<tr>
<td>9: else</td>
</tr>
<tr>
<td>10: remove \textit{(s,t)} from \textit{derivedMappings};</td>
</tr>
<tr>
<td>11: if \textit{t} is child of \textit{s’} or \textit{s’} is child of \textit{s} or \textit{s} is sibling of \textit{s’}</td>
</tr>
<tr>
<td>12: if \textit{s} is ranked ahead of \textit{s’} corresponding to \textit{t} in by column strategy then</td>
</tr>
<tr>
<td>13: remove \textit{(s’,t’)} from \textit{derivedMappings};</td>
</tr>
<tr>
<td>14: else</td>
</tr>
<tr>
<td>15: remove \textit{(s,t)} from \textit{derivedMappings};</td>
</tr>
<tr>
<td>16: return \textit{derivedMappings};</td>
</tr>
</tbody>
</table>

5. Experiments and Analysis

In order to evaluate our uncertain schema matching algorithm, we construct 5 schema matchers each generates a similarity matrix as its matching result and a composite matcher which combines matching results of these schema matchers. We use the name of schema element, structure information of schema and thesauri as heuristics and each schema matcher use particular algorithms to compute similarities of schema element pair. In our experiments, we use two XML purchase orders, CIDX and Excel, from www.BizTalk.org [8] to evaluate our uncertain schema matching algorithm. We use mappings selected by domain expert as the accurate mappings and compare matching results with these accurate mappings. We choose these particular schemas because they contain typical heterogeneities of data schemas such as differences of names, data types and structures. And the accurate mappings selected by domain experts contain non-matching elements and 1:n mappings. We employ Precision, Recall, Overall and F-Measure as the matching quality measurements. According to the definition of these measurements, schema matchers should try their best to identify true positive candidate mappings and filter out false positive candidate mappings. As demonstrated in [16], neither Precision nor Recall alone can assess the quality of schema matching algorithms roundly. In particular, Recall can easily be maximized at the expense of a poor Precision by returning all possible correspondences. On the other hand, a high Precision can be achieved at the expense of a poor Recall by returning only few correct correspondences. Hence it is necessary to consider both measures at the same time. Overall and F-Measure are combined measures and F-Measure is a compromise of Precision and Recall.

We compare Interval Fuzzy Ranking algorithm with two categories of algorithm. The first category, Average Threshold (AT for short) algorithm, uses average operator to aggregate matching results of different matchers and uses threshold strategy to select candidate mappings. The second category, Average Top-K (ATK for short) algorithm, uses average operator to aggregate matching results of different matchers and uses top-k strategy to select candidate mappings. First we compare AT with different thresholds in Figure 1 and choose a threshold for AT corresponding to its best performance. We compare ATK with different k values in Figure 2 and choose a k value for ATK corresponding to its best performance. We demonstrate from Figure 1 and Figure 2 that it is difficult to choose a proper threshold in AT and k value in ATK. Then we compare IFR with and without semantic conflict elimination in Figure 3 which shows that a semantic conflict elimination step is essential to remove false positive mappings. Finally, we compare IFR with AT and ATK in Figure 4 which demonstrates that IFR performs better AT and ATK.
Figure 1 illustrates that the matching quality of the first category of algorithm fluctuate with different thresholds. As shown in Figure 1, we can achieve a high Recall when we choose 0.1 as the threshold. But the threshold of 0.1 will result in a low Precision. In the same way, we can achieve a high Precision when we choose 0.9 as the threshold. But the threshold of 0.9 will result in a low Recall. It is difficult to balance between Precision and Recall. In order to obtain a relatively high matching quality, 0.7 is most probably chosen as the threshold. Figure 2 illustrates that the matching quality of the second category of algorithm fluctuate with different $k$ values. As demonstrated in Figure 2, we can achieve a high Precision by choosing 15 as the $k$ value. But the $k$ value of 15 will result in a low Recall. And we can achieve a high Recall by choosing 65 as the $k$ value. But the Precision corresponding to the $k$ value of 65 is low. According to the Precision curve, there are false positives even in top-5 candidate mappings. And 40 is most probably chosen as the $k$ value to achieve a relatively high matching quality. As shown both in Figure 1 and Figure 2, F-Measure is a compromise of Precision and Recall, and the curve of F-Measure is always between Precision and Recall curve while Overall is more pessimistic than F-Measure.
Figure 3. Comparison of IFR with and without semantic conflict elimination

Figure 3 illustrates the comparison of IFR with and without semantic conflict elimination. As shown in Figure 3, IFR algorithm both with and without semantic conflict elimination get the Recall measurement of 1.0 which means that candidate mapping selection procedure can identify all true positive candidate mappings. While Precision, Overall and F-Measure have a remarkable improvement which means semantic conflict elimination procedure can remove false positive candidate mappings effectively. So a semantic conflict elimination step is essential to achieve high quality matching results.

Figure 4. Comparison of AT, ATK and IFR algorithm

Figure 4 illustrates the matching quality of IFR algorithm compared with AT and ATK. As shown in Figure 4, the qualities of AT and ATK are relatively low and not in equilibrium while the quality of IFR is relatively high and balanced. Compared with former two algorithms, we can achieve a high F-Measure of 0.98 which is a compromise of Precision and Recall, while both of former two algorithms can only achieve a 0.73 F-Measure. Both AT and ATK compare all candidate mappings at the same time. And as demonstrated in [11] that candidate mappings which are ranked high are not always accurate mappings. While in iterations of IFR only candidate mappings correspond to one schema element are compared. But this will lead to the selection of candidate mappings containing schema element which is not mapped to any schema element. In order to avoid this case, IFR uses cross comparison to confirm identifications, i.e. employ by row strategy and by column strategy at the same time. In addition, IFR is designed to identify 1:n mappings while AT and ATK only indentify 1:n
mappings unconsciously, i.e. they identify 1:n mappings as long as the similarities of these candidate mappings are high enough.

6. Conclusions

In this paper, a framework for managing uncertainty of schema matching is proposed. Based on the concept of composite matcher, we take advantage of results of independent schema matchers. First, we construct interval fuzzy similarities from these results and rank candidate mappings by their interval fuzzy similarities. Then we select candidate mappings by a well designed mechanism. At last, a semantic conflict elimination procedure is executed to remove false positive candidate mappings. The advantage of IFR algorithm is that it need not to choose a threshold or k value, and it can identify both 1:1 and 1:n mappings. Experiments have demonstrated that our algorithm can identify true positive candidate mappings and filter out false positive candidate mappings effectively.

7. References