Restricting the Overlap of Top-N Sets in Schema Matching

Eric Peukert
SAP Research Dresden
01187 Dresden, Germany
eric.peukert@sap.com

Erhard Rahm
University of Leipzig
Leipzig, Germany
rahm@informatik.uni-leipzig.de

ABSTRACT
Computing similarities between metadata elements is an essential process in schema and ontology matching systems. Such systems aim at reducing the manual effort of finding mappings for data integration or ontology alignment. Similarity measures compute syntactic, semantic or structural similarities of metadata elements. Typically, different similarities are combined and the most similar element pairs are selected to produce a best-1 mapping suggestion.
Unfortunately automatic schema matching systems are only rarely commercially adopted since correcting the best-1mapping suggestion is often harder than finding the mapping manually. To alleviate this, schema matching must be used incrementally by computing Top-N mapping suggestions that the user can select from. However, current similarity measures and selection operators suggest the same target elements for many different source elements. This effect, that we call overlap, reduces the quality of schema matching significantly.
To address this problem, we introduce a new weighted token similarity measure that implicitly decreases the overlap between Top-N sets. Secondly, a new Top-N selection operator is introduced that is able to increase the recall by restricting overlap directly. We evaluate our approaches on large-sized, real world matching problems and show the positive effect on match quality.

Categories and Subject Descriptors
D.2.12 [**Interoperability**]: Data mapping

General Terms
Algorithms

Keywords
Top-N, matching, incremental matching, weighted name matching, overlap

1. INTRODUCTION
Finding mappings between complex metadata structures is a time-consuming task in a number of fields such as data integration or ontology alignment [14]. In order to speed up that process schema matching systems compute a mapping suggestion that ideally is close to the intended mapping [18][16]. Such systems apply a number of similarity measures to compute syntactic, semantic or structural similarities between metadata elements. These similarities are combined using dedicated similarity combination techniques [15]. Finally a selection step [13][4] extracts the most probable element pairs from the combined similarities and constructs the final mapping suggestion.

Unfortunately when using state-of-the-art schema- or ontology matching prototypes from research [11][9][5] for matching large business schemas the quality of the mapping suggestion is often below being usable. While published match results exhibit mostly FMeasure values of more than 0.8, we observed values below 0.4 for complex business schemas. The effort to correct the computed mapping in such cases is often much higher than doing the whole mapping manually. Also business users often do not trust automatically computed mappings since they do not overview possible ambiguities and only the best-1 element suggestion is visible.

In this paper we propose to apply schema matching incrementally that lets the user select source elements individually. For each selected source element, N mapping target element suggestions are computed (e.g. N=3-10) that we call Top-N sets. The user then needs to select the correct target out of the individual Top-N sets. Compared to the best-1 approach this slightly increases the effort for the user. However we show that this approach makes schema matching usable in practice. Our observation is, that the probability that the correct target element is within a Top-10 set is sometimes bigger than 0.9 for business schema mapping problems. Thus with high probability the user can select out of 10 target elements instead of looking up the whole target schema. This also significantly reduces the mapping effort.

Some matching systems support Top-N selection, e.g. COMA++ with a so-called maxN selection [4]. However, the computed Top-N results reveal a big problem: The Top-N sets for different source elements can highly overlap, especially for business schemas. This means, that some target elements appear in Top-N sets of multiple or many source elements. These overlapping elements reduce the space for potentially correct matching target elements. One reason is that business schema element names often consist of multiple words and some are more frequently used within a schema than others. This leads to wrong target matches and overlap for different source elements that are similar only due to some frequent word. Secondly the selection operator maxN creates Top-N sets without considering or restricting the produced overlap. We project, that reducing the overlap increases the probability to have a correct target element in Top-N sets.

Our contributions are the following:

1. We propose a new weighted token-based name matcher that is able to reduce the ambiguities produced by frequent tokens.
2. We analyze the problem of overlap in Top-N sets and propose a selection algorithm that is able to compute Top-N sets with restricted overlap.

NTSS 2011, Mar 25, 2011, Uppsala, Sweden.

3. We evaluate our finding by mapping real world business schemas. We can show that our weighted token-based name matcher improves match quality. Moreover by reducing the overlap the recall can be improved on a number of mapping problems.

Section 2 analyzes the problem of overlap in detail. Section 3 and 4 introduce our similarity measure and the new Top-N overlap selection operator. In Section 5 a broad evaluation of both approaches is presented. Finally we review related work and close in Section 7 with an outlook.

2. DEFINITIONS & PROBLEMS

Before describing the problem in detail, we first introduce some schema matching basics.

A schema *S* consists of a list of schema elements s with $s \in S$. Each schema element *s* has a name *n*, one or no parent schema element, and a set of children schema elements. Schema elements can carry additional information such as data type, description (documentation) or cardinality. The schema type is generic and refers to any metadata structure that can be matched such as trees, ontologies, meta models, as well as database schemas.

A *mapping M* between a source schema *S* and target schema *T* is a set of correspondences. Each correspondence $c \in M$ links an element of the source schema s_i to an element t_i of the target schema: $c := (s_i, t_i)$. Figure 1 shows a running example with related schemas S1 and S2 describing personal data. Each element gets a capital letter for easier reference. A correct mapping between S1 and S2 would contain the shown correspondences (S1.contactPhone, S2.telephone), (S1.contactName, S2.contactNData) and (S1.colorNdata, S2.eyecolor).

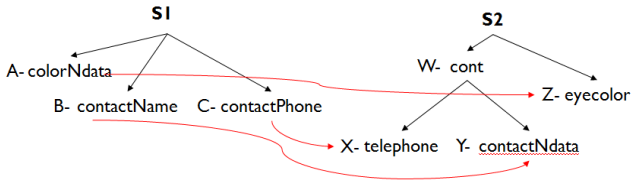


Figure 1: Running Example

Mappings are computed with the help of similarity matrices. A similarity matrix $A = a_{i,j}$ consists of |S| * |T| cells. |S| and |T| are the numbers of schema elements of the source and target schema. The matrix has |S| rows and |T| columns. Each cell $a_{i,j}$ contains a similarity value between 0 and 1 representing the similarity between the *i*-th element of the source schema and the *j*-th element of the target schema. Low similarity values close to 0 indicate non-matching element pairs. Each *matching algorithm* is assumed to return for input schemas *S* and *T* a similarity matrix: A = *mat*(S,T).

Several such matrices $A_1 \ldots A_x$ referring to the same source and target schemas can be combined to a new matrix *B* using some combination function *f, e.g. by taking the average similarity values* [15]. The final correspondences can be determined by a *selection* on a matrix *A* that applies a condition *c* on each cell. If *c* evaluates to false, the value of the cell is set to 0 to indicate a non-matching element pair; otherwise $b_{i,j}=a_{i,j}$. Different conditions can be used for c such as: Threshold, Delta or maxN [4].

In this paper, we aim at computing multiple target element suggestions for each source element. For that reason, we introduce a so called *Top-N mapping*. A Top-N mapping Q consists of a set of extended correspondences $(s_i, \{t_{i1} \ldots t_{iN}\}) \in Q$ where a source element $s_i \in S$ is linked to a set of at most *N* target elements that likely correspond to s_i. These sets are called Top-N sets. Top-N sets do not need to be filled with N target elements if there are less than N target elements with sufficient similarity.

2.1 Word token ambiguity

Common name matchers, e.g. based on n-gram string similarity or token overlap, have problems to correctly identify matching elements in the presence of reoccurring tokens, such as **contact** in **S1**. Since most schema- and ontology matching systems heavily rely on name similarity, we analyze the current behavior of a name matcher in detail. Figure 2 shows the process of the name matcher of [4].

It computes the similarity of two words by first tokenizing (1) the elements. For each element pair the elements are put into a token similarity matrix (2) consisting of the source and target tokens and a similarity value computed by applying some string similarity measure like trigram or edit-distance.

1. tokenize

element	tokens
contactName	contact, name
contactNdata	contact, ndata
telephone	telephone
…	…

2. compute token similarity

C-X	telephone
contact	.1
phone	.55

C-Y	contact	ndata
contact	1	0
phone	0	0

3. compute set similarity & filter top N targets

	W	X	Y	Z
A	.20	0	.62	.39
B	.44	.11	.57	.08
C	.44	.4	.5	.07

Figure 2: Name matcher process

Finally (3) the set similarity measure is computed by

$$setSim = \frac{\sum_i max_{i,j} + \sum_j max_{i,j}}{c+r} \quad (1)$$

Where *i,j* are indexes of a token similarity matrix with *r* rows and *c* columns. The set similarity collects the maximum values of each row and column and divides it by the sum of column and row cells. In the example, the set similarity for the **C-X** token matrix computes as $\text{setSim}_{C-X} = \frac{0.1 + 0.55 + 0.55}{2+1} = 0.4$.

The final similarity matrix can then be used as input to structural matchers. It can be combined with other mappings or a selection can be applied to compute a final mapping. In the example of Figure 2 we applied a max-2 selection strategy that chooses the maximum 2 target elements (the Top-2 sets) for each source element. They are marked in grey for each row. The correct matches are marked by black boxes. Obviously due to the high similarity of **contactPhone(C)** and **contactNdata(Y)** the correct match of **contactPhone(C)** to **telephone(X)** is not included in the Top-2 set of **contactPhone(C)**. The frequent token **contact** distorts the result and should be considered as less important when computing the name similarities.

2.2 Overlap of Top-N sets

An observation that can be made for the sample similarity matrix in Fig. 2 is, that the target elements contained in different Top-N

sets of the Top-N mapping overlap strongly. The Top-N sets for **B** and **C** are similar. Moreover, element **Y** is contained in all three Top-N sets. Thus the overlap for that element is very high. Obviously, it is highly unlikely that every source item should match to the same target item. In the example, this was introduced by the ambiguity imposed by the token **contact**. We will show in the evaluation that when matching business schemas the Top-N sets generated by applying the selection strategy max-N are also strongly overlapping. In some cases target elements appeared in up to 70 source element Top-N sets. This fills up these Top-N sets so that correct matches might not be included. In order to measure the overlap we first need to give a definition of the overlap.

Definition of Overlap: Given a Top-N mapping Q between a source schema S and a target schema T that consist of up to $|S|$ Top-N sets *P*. The overlap *overlap(t)* of a target element $t \in T$, is the number of Top-N sets it is contained in.

Our assumption is, that a restriction of the overlap of target elements in a Top-N mapping improves the recall. Since we cope with Top-N results we also need to measure a recall over Top N similar to information retrieval. *Recall over Top-N* measures the ratio of the number of Top-N sets that contain a correct target element to the total number of matching target elements. In the example of Figure 2 the overlap for target element **Y** is 3 and for element **W** the overlap is 2. The recall over Top-2 is 0.67.

Note that we could also define a precision over Top-N that computes ratio of Top-N sets that contain a correct result. Also the rank of correct results within Top-N sets could be included to weight individual correct sets.

3. WEIGHTED TOKEN NAME MATCHER

In order to cope with the ambiguity and overlap produced by frequent tokens we propose to extend the name matcher by a token weighting mechanism. We need to find an appropriate weighting function for tokens and need to include the computed weights into the name matcher algorithm. For that purpose the popular *term frequency – inverse document frequency* (TF-IDF) approach from Information Retrieval is adapted. It assigns a weight *wT* to a term *t* in a document *d* from all documents *D*. The term frequency *tf* computes the ratio of the number of occurrences $freq_{t,d}$ of term *t* in *d* and the maximal frequency $max_a(freq_{a,d})$ over all terms within the document. It measures how important a term is for a specific document. In contrast to that, the inverse document frequency *idf* computes the overall importance of a term for the whole document corpus.

$$wT_{t,d} = tf_{t,d} * idf_t = \frac{freq_{t,d}}{max_a(freq_{a,d})} * \log\left(\frac{|D|}{n_t}\right) \quad (2)$$

In the schema matching context we adapt TF-IDF so that our documents are the element names to be matched and the terms are the name-tokens after the tokenization phase. The formula can easily be translated to *token frequency - inverse element frequency*. With *k* being a token and *e* an element name from the schema.

$$wtok_{k,e} = tokf_{k,e} * ief_k \quad (3)$$

Since a token in most cases occurs only once in an element name the $tokf_{k,e}$ -part reduces to one: $tf_{t,d} \rightarrow tokf_{k,e} = 1$. The inverse element frequency is based on the number of schema elements N and the occurrence count c_k of the *k*-th token of all tokens from all schema elements: $idf_t \rightarrow ief_k = \log\left(\frac{N}{c_k}\right)$. Thus the computation of the individual weight of a token *k* when normalized to the maximum possible value (the weight is between [0,1]) is reduced to

$$w_k = \frac{\log\left(\frac{N}{c_k}\right)}{\log\left(\frac{N}{1}\right)} \quad (4)$$

The individual weight w_k for a token *k* will be used when computing the token similarities. For each token-pair the weights of the source and the target token need to be combined to a combined weight: $w_{i,j} = w_i * w_j$.

Now, we can use $w_{i,j}$ to weight token similarities for individual element comparisons. That means that token similarities for less frequent tokens will be increased whereas similarities for pairs with often occurring tokens are decreased. We adapted the set similarity computation so that the maximum values of rows and columns are weighted and normalized by the sum of weights of each maximum value. This ensures that within an element comparison the influence of frequent tokens is decreased while at the same time the influence of less frequent tokens increases.

$$setSimWeighted = \frac{\sum_i w_{i,j}max_{i,j} + \sum_j w_{i,j}max_{i,j}}{\sum w_{i,j}} \quad (5)$$

In order to better explain the new token weighting, the process is described along the running example from above (see Figure 3). As with the original name matcher first the names are tokenized (1). Then the occurrences of tokens are counted (2). The token similarities are computed for each element comparison and put into the token similarity matrices (3). Based on the occurrence counts, the combined token weights are computed for each token similarity matrix entry (4). Then the new set similarity is computed for each token similarity matrix (5). The results are put into the name similarity matrix (6). When selecting the Top-2 target values for each source element we can see that the overlap has reduced. The **C**-**X**-value is now correctly within the Top-2 set of **C** and is also the best match for **C**. Also we can see that the similarities of target elements to **B** decreased. The reason is, that all these name-comparisons involve the frequent term **contact** which is considered less important by the new weighted name matcher. The Recall over Top-N is now one since all correct correspondences are included in the Top-2 mapping.

The new weighted name matcher can now be used as constituent matcher for structural matchers (e.g. Name-Path Matcher) to also improve their results. This will also be shown in the evaluations.

4. TOP-N SELECTION WITH RESTRICTED OVERLAP

Still the new weighted token-based name matcher might produce significant overlap in its result. We therefore propose a further approach to restrict the overlap within the final selection of a matching system. For that purpose the Top-N overlap problem is formalized.

We are given a Top-N mapping Q consisting of a number of Top-N sets P. Additionally we introduce a constraint α that restricts the maximum overlap of target elements in a Top-N mapping. The goal of Top-N set overlap reduction is to maximize the sum $\sum_{\{p \in P\}} \sum_{\{a \in p\}} a$ of target similarity values within the Top-N sets P while staying within the bounds of the given overlap value: $\vee a \in p$: overlap(a) $\leq \alpha$ for each $p \in P$.

With bigger N the possible overlap increases. If the target schema is smaller than the source schema then α will always be bigger than 1. Thus the order of matching influences the overlap.

1. Tokenize

element	tokens
contactName	contact, name
contactNdata	contact, ndata
telephone	telephone
…	…

2. Count occurrence

s-token	
contact	2
name	1
phone	1
…	

t-token	
cont	1
telephone	1
contact	1
…	

3. Compute token similarity

C-Y	contact	ndata
contact	1	0
phone	0	0

C-X	telephone
contact	.1
phone	.55

4. Compute token weights

W_{C-Y}	contact	ndata
contact	.36	.36
phone	1	1

W_{C-X}	contact
contact	.36
phone	1

5. Compute weighted set similarity

s(C,Y)= (0.36*1 + 0*1 + 0.36*1 + 0*1)/(0.36+1+ 0.36+1= 0.26

s(C,X)= (0.11*0.36 + 0.55*1 + 0.55*1)/(0.36+1+1)= 0.48

6. Filter top N targets

	W	X	Y	Z
A	.20	0	.62	.39
B	.28	.11	.38	.1
C	.28	.48	.26	.09

Figure 3: Weighted name matcher process

If there is a choice, then taking the smaller schema as source is recommended, since less Top-N sets with lower overlap will be computed. However, for business schemata where data transformations are needed, the source and target are fixed.

In order to find optimal Top-N sets with a naïve approach all possible combinations of Top-N sets would need to be generated and compared. For each row of a similarity matrix with |T| columns all subsets of size N need to be generated. Since we have |S| rows, the number of combinations of Top-N sets is the following:

$$combinations = \binom{|T|}{N}^{|S|} \quad (7)$$

Finding the optimal Top-N set assignment that restricts the overlap to a given value is at least in NP since it is easy to check weather a combination is a valid combination or the maximum combination visited yet. We reviewed related problems. First, Stable Marriage [12] can be seen as an instance of the Top-1 overlap problem. It tries to find an optimal 1:1 assignment between source and target elements while fixing the maximum overlap for target elements to α=1. The resulting solution imposes a very restrictive 1-overlap. If the schemata are not equally sized, not every source element finds a partner. The Top-N overlap problem can also be mapped to more general problems such as General Assignment or Knapsack [10]. What we need is a solution to "a multiple choice multiple knapsack" problem that we could not find in literature.

Since finding the optimal solution is problematic we propose a heuristic algorithm that approximates a Top- N set with restricted overlap fastly. The computed sets are not optimal but – as our evaluation will show – they already improve recall for real world mapping problems. Alternative approaches to further improve matching quality are left for future research.

4.1 Basic Top-N overlap restriction algorithm

In Algorithm 1 an algorithm is introduced that computes a solution to the Top-N-set overlap problem.

The input to the algorithm is a similarity matrix A, the size of Top-N-sets and the overlap constraint α to restrict the maximal overlap per target element. In lines 2 and 3, the columns and rows of the matrix are sorted according to the similarity values and the initial Top-N sets are computed using an existing maxN selection algorithm. In lines 4 and 5 the target element (column) with the maximum overlap is identified. This requires computing the overlap for each target element. In the algorithm part from line 7 to 17 it is tried to iteratively reduce the overlap for the maximally overlapping column by increasing another column's overlap. The algorithm is reducing the overlap counts until α or the minimal possible overlap is reached. On lines 9 to 11 an entry to remove from a Top-N set is identified. On lines 12 to 15 a replacement for the entry to remove is identified. In lines 16 and 17 the Top-N sets are changed accordingly. Finally in line 18 the resulting Top-N sets are returned.

```
1  Input: similarity matrix A; N, Alpha
2  sortRowsAndColumns(A)
3  topNSets=compute the Top-N sets for each source element
4  compute overlap counts for each target element
5  maxColumn=choose target column with max overlap
6  found=true
7  while overlap(maxColumn) >Alpha and found=true
8        found=false, toRemove=null
9        iterate each columnEntry of maxColumn ascending order
10             if columnEntry is in Top-N set
11                   toRemove= columnEntry, break;
12       iterate each rowEntry of row of toRemove descending order
13             if rowEntry is not in Top-N set
14                   toAdd = rowEntry
15                   found=true, break;
16       topNsets.remove(toRemove)
17       topNsets.add(toAdd), recompute overlap counts
18 return  topNSets
```

Algorithm 1: Basic Algorithm for finding a Top-N set overlap solution

In Figure 4: Running Algorithm the algorithm is applied for the Top-2 mapping from Figure 2 with an alpha-value of 2.

(a)

	W	X	Y	Z
A	.20	0	.62	.39
B	.44	.11	.57	.08
C	.44	.4	.5	.07
	2	0	3	1

(b)

	W	X	Y	Z
A	.20	0	.62	.39
B	.44	.11	.57	.08
C	.44	.4	~~.5~~	.07
	2	1	2	1

Figure 4: Running Algorithm

The overlap is computed for each target element (a) and **Y** is chosen as **maxColumn** exceeding the alpha threshold. Then the minimal element from the **Y**-column is selected as **toRemove** element which is **C-Y**. Then the row **C-Y** is iterated to find the

next candidate (**C-X**) that can be added to a Top-N set. **C-Y** is removed and **C-X** is added. The overlap counts are recalculated. Since no overlap count is bigger than alpha, the algorithm terminates and returns the Top-N set from (b). The final recall over Top-N is now 1 since all correct target elements are within the Top-N sets. In general, the algorithm can miss the optimal solution since the columns are processed in a fixed order.However in our later experiments we show that our solution is already able to achieve significant improvements on the recall over Top-N.

5. EVALUATION

Our evaluation is based on real world business schema mapping problems. We first want to prove the existence of the overlap problem. We then evaluate the new weighted token name matcher and the Top-N overlap restriction algorithm.

For evaluation of the quality of the Top-N incremental mapping approach much more emphasis must be set on recall. The best known quality measures like FMeasure and Overall do measure how close the automatically computed mapping is to the correct mapping.

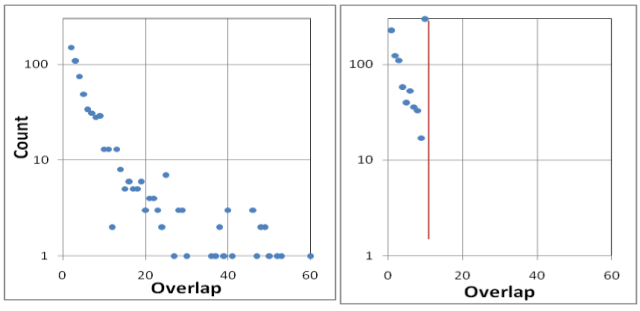
5.1 Test data

For evaluating the new Weighted Name Matcher and the Top-N overlap selection strategy, we took a set of real world business to business schema mappings. These schema mappings consist of mappings between invoices, orders and other business objects. Some scenarios cope with mapping complex EDIFACT schemata others need to cope with SAP IDOCs, and XSDs from the SAP Enterprise Services Repository. The schema sizes range from 100 to 4500 elements per schema. Some schemas contain documentation whereas others have cryptic element names. Please see the table below that characterizes the data set. #M refers to the number of mapping testcases that are contained in a data set.

Enterprise Services	#M	Features
SRM	8	Long readable names
RFID	10	IDOC with cryptic names & XSDs with long names
CRM	8	IDOC & XSDs with long names
FINANCIAL	4	Short readable names
EDUCATION	7	Short readable names
AUTOMOTIVE	2	Multi token names, deep paths

5.2 TOP-N Overlap restriction

Initially we want to verify the existence of the overlap problem and show the effect of overlap reduction.



(a) (b)

Figure 5 Distribution of overlap before and after overlap restriction

We took a large exemplary mapping scenario from our AUTOMOTIVE data set that maps a delivery schedule message (DELFOR) for electronic data interchange (EDI) between trading partners to the Joint Automotive Industry standard. The correct mapping consists of 450 correspondences. We used a default matching configuration with a token path and a leaf matcher and applied the classical maxN selection with N=5. A recall of 0.56 is achieved. Figure 5 (a) visualizes the distribution of overlap counts for the computed mapping. Obviously a number of target elements appear up to 60 times in different Top-N sets. By applying our new Top-N overlap selection approach we can restrict the overlap to 10. Figure 5 (b) shows the new distribution. The overlap is never bigger than 10. However, more than 100 elements still overlap Top-N sets 10 times. As expected the overlap reduction increased the recall which is now at 0.6.

5.3 Aggregated Results

We first evaluate the quality of the weighted token name matcher. For that purpose we compare its results with the classical name-matcher. Since the name matcher is often the base matcher for structural matchers we compare a classical name-path matcher with a new weighted name path that is using the new name matcher implementation. We ran all following matching tasks on the whole data-set and averaged the resulting recall values. Figure 6 visualizes the aggregated results for Top-5.

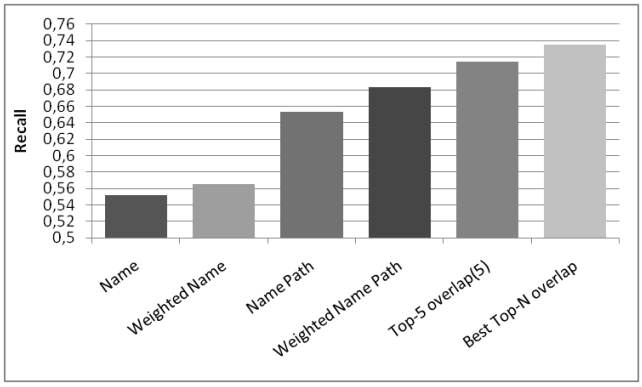


Figure 6 Comparing all newly introduced strategies

We are able to achieve a small but noticeable improvement in the recall compared to the classical name matcher. This improvement can become larger for dependent structural matchers. As shown in Figure 6 the name path matcher using the weighted name matcher as base matcher improved by nearly five percent. Our observation is that the new name matcher should not be applied in all cases. For schemas with many recurring schema parts the token frequencies for some tokens are high. However, reducing the weights for those tokens would be misleading. If more than 40 percent of a schema consists of reoccurring components the classical name matcher should be applied.

In a next run we evaluate the new Top-N overlap selection approach in comparison with the maxN selection. As overlap value we use 5. The new algorithm only needed few seconds to compute the Top-N mappings even for the large test cases. On average, our new selection algorithm increases the recall by another five percent (Figure 6), which sums up to a 10 percent increase of the recall. However, there is still room for improvement. When setting the optimal overlap value for each test case an average recall of nearly 0.74 could be reached. In Figure 6 this is shown as the Best Top-N overlap. Finding the

overlap value dynamically based on the schema features is up to future work. However, taking the value N also for the overlap value turned out to be a good choice. Also choosing an overlap that is higher than the minimal possible overlap is reasonable. We also investigated whether this effect holds for different values of N. Figure 7 compares the weighted name path approach and the Top-N overlap approach with different Top N overlap selections. Obviously, the effects also hold for N=1 and bigger N.

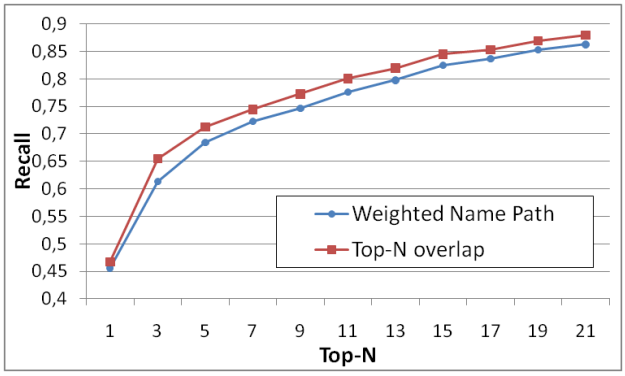


Figure 7: Comparing Weighting approaches in for different top-N selections

In practice, N needs to be chosen by the user. With bigger N the bigger the chances are that correct mappings are inclueded in the Top-N sets. However, this also increases the effort for the user. If all similar target elements are included in top-N sets, there is nothing to influence by the overlap restriction.

6. RELATED WORK

Other researchers from schema matching already considered that selecting the top N matching elements is promising. Most systems offer a selection strategy that can cope with 1:N matches [4][17]. Usually maxN with N=1 or N=2 returned good results. However the maxN-selection does not leverage interdependencies of Top-N sets or restrict the overlap. COMA++ [5] supports a selection direction BOTH that evaluates all columns when computing the Top-N sets for source elements. However, it strives for a stable marriage property which does not allow any overlap. Also the resulting Top-N sets do often contain much less than N elements resulting in lower recall. This is accepted to achieve a high precision for the best-1 mapping suggestion.

Recently some work was introduced that tries to leverage multiple mapping solutions for computing a match result [3][6][8]. One approach [8] is similarly called Top-K. The major distinction to our work is their goal of finding a Top-1 mapping. They internally compute K competing mapping solutions. A heuristic then selects the best mapping based on a stability analysis of the K mappings. This improves precision at the cost of recall. In our work we want to present the user the Top-N target elements for a source element with restricted overlap. In our heuristic, target elements that often occur in Top-N sets should be reduced to a defined overlap-constraint.

Some work was proposed on incremental matching [1][2] that includes user feedback into the computation of further Top-N sets [2]. It is related since an already chosen target match should not occur again in a subsequent top-N set. In our words, the overlap for that specific match is set to 1. Also learning-based solutions were proposed [7] that take first user selected target elements as gold standard for learning-based algorithms.

7. Conclusion and Outlook

In this paper we propose to change the way automatic schema matching is currently applied. Instead of the best-1 mapping solution we present the user with the Top-N solution that she can select from element by element. We identified an overlap problem for such Top-N correspondences and proposed a first approach to limit the overlap for improved recall. We also introduced a new name matcher approach that is able to weight tokens depending on their occurrence counts.

Our evaluations demonstrate the value of the new approaches. In sum the Top-N overlap selection techniques and the new name matcher add around 10 percent improvement on recall without adding a new matcher to a schema matching system. Still there is room for further improvement. First, the Top-N overlap restriction algorithm does not find the optimal solution. We discussed some probable solution paths that could be followed in future research. In the evaluation it turned out that preprocessing and feature based matcher selection is needed, that will be generalized in future.

8. REFERENCES

[1] Bernstein, P. A., Melnik, S., Churchill, J. E., 2006. Incremental schema matching. VLDB.

[2] Cao, Z.,Li, K., Liu, 2009. Putting Feedback into Incremental Schema Matching. WCSE.

[3] Cheng, R., Gong, J., Cheung, D. 2010. Managing Uncertainty of XML Schema Matching. ICDE .

[4] Do , H. H. and Rahm, E. 2006. COMA - A System for Flexible Combination of Matching Approaches. VLDB Proc.

[5] Do, H. H. and Rahm, E. 2007. Matching large schemas: Approaches and evaluation. Inf. Syst., 32(6).

[6] Dong, X., Halevy, A. Y., Yu, C. 2007. Data integration with uncertainty. VLDB Proc.

[7] Ehrig,M., Staab, S., Sure. Y. 2005. Bootstrapping Ontology Alignment Methods with APFEL. WWW.

[8] Gal, A. 2006. Managing Uncertainty in Schema Matching with Top-K Schema Mappings. J. on Data Semantics VI.

[9] Hu, W. and Qu, Y. 2008. Falcon-AO: A practical ontology matching system. Web Semant., 6(3).

[10] Kellerer, H., et. al., *Knapsack Problems* , 2005. Springer

[11] Li, J. et al. 2009. RiMOM: A Dynamic Multistrategy Ontology Alignment Framework. TKD 21(8).

[12] Marie, A., Gal, A.,2007. On the Stable Marriage of Maximum Weight Royal Couples. II Web Workshop.

[13] Meilicke, C., Stuckenschmidt, H. 2007. Analyzing Mapping Extraction Approaches. ISWC - Workshop on Ontology Matching

[14] Noy, N. F. and Musen, M. A. 2003. The PROMPT Suite:Interactive Tools for Ontology Merging and Mapping. Int. J. Hum.-Comput. Stud.

[15] Peukert, E., Maßmann, S., König, K. 2010. Comparing Similarity Combination Methods for Schema Matching GI-Workshop - Informationsintegration in Service-Architekturen

[16] Rahm, E. and Bernstein, P. A. 2007. A survey of approaches to automatic schema matching. The VLDB Journal 10.

[17] Roitman, H., Gal, A., Domshlak, C., 2008. Providing Top-K Alternative Schema Matchings with Onto Matcher. ER.

[18] Shvaiko, P and Euzenat J. 2005. A Survey of Schema-Based Matching Approaches. Journal on Data Semantics IV