UFeed: Refining Web Data Integration Based on User Feedback

Ahmed El-Roby
University of Waterloo
aelroby@uwaterloo.ca

Ashraf Aboulnaga
Qatar Computing Research Institute, HBKU
aaboulnaga@hbku.edu.qa

ABSTRACT

One of the main challenges in large-scale data integration for relational schemas is creating an accurate mediated schema, and generating accurate semantic mappings between heterogeneous data sources and this mediated schema. Some applications can start with a moderately accurate mediated schema and mappings and refine them over time, which is referred to as the pay-as-you-go approach to data integration. Creating the mediated schema and mappings automatically to bootstrap the pay-as-you-go approach has been extensively studied. However, refining the mediated schema and mappings is still an open challenge because the data sources are usually heterogeneous and use diverse and sometimes ambiguous vocabularies. In this paper, we introduce UFeed, a system that refines relational mediated schemas and mappings based on user feedback over query answers. UFeed translates user actions into refinement operations that are applied to the mediated schema and mappings to improve their quality. We experimentally verify that UFeed improves the quality of query answers over real heterogeneous data sources extracted from the web.

1 INTRODUCTION

The web contains a massive amount of crawlable, table-structured data. This data is extracted from various sources, such as HTML tables, web forms, and on-line spreadsheets [1, 8, 16]. Each of these data sources can be viewed as a *relational data source* (i.e., a table with a schema). The number of such sources is very large and continuously increasing. For example, 125M HTML tables were extracted in 2015 [15].

Users and application programs can greatly benefit from having a unified interface to simultaneously query multiple of these data sources. Data integration systems for relational data provide such a unified interface by automatically building a relational mediated schema for the data sources along with semantic mappings between the schemas of the data sources and the mediated schema. Despite substantial research progress in the field of data integration, web data sources such as the ones mentioned above still represent a significant challenge for traditional data integration. One reason is the scale of the problem, since web data integration typically deals with a large number of data sources. The more important reason is that these data sources are semantically heterogeneous to a high degree. Data sources on the web deal with many different real world domains (e.g., sports, movies, finance, etc.), and the representation of table names, attribute names, and data values is based on human language and can therefore be ambiguous, inconsistent, and varying, even if the data sources are clustered into individual domains [21].

A good way to address the challenge of data integration on the web is to recognize that many applications do not require full integration of the data sources that they use in order to provide useful services. This encourages a *pay-as-you-go* approach to integrating web data sources [17]. The pay-as-you-go approach involves two phases: *setup* and *refinement*. In the setup phase, the system creates: (1) a mediated schema or possibly several schemas, and (2) mappings between the schemas of the data sources and the mediated schema(s). Since setup is done fully automatically, the mediated schema and mappings will likely contain semantic errors. The pay-as-you-go philosophy requires the mediated schema and mappings to be *refined* during use based on feedback from the user. The typical way a user would use the mediated schema is to issue queries against it and receive answers to these queries from the data sources. Thus, the natural way for a user to provide feedback is to indicate the correctness of the tuples that she sees in the answers to her queries. Surprisingly, most of the prior work on the refinement step has been decoupled from the querying process. Instead of requiring a user to provide feedback on the answers to her queries, most prior refinement approaches expose the user to the details of the mediated schema and the source schemas used to answer her queries, and enable her to directly modify the mediated schema and mappings. Thus, instead of the system automatically fixing its mistakes based on feedback from the user, the system presents these mistakes to the user and asks her to fix them. Such approaches implicitly assume that the user is a database expert, which may not be true in practice. In addition, the focus in prior work has been on refining the mappings, assuming that the target schema (mediated schema) is correct. While this assumption is valid when the mediated schema is manually created (and therefore of high quality), this is not a practical assumption for pay-as-you-go data integration, where the mediated schema is automatically generated.

This paper introduces UFeed, a pay-as-you-go data integration system that addresses the problems with prior refinement approaches. To the best of our knowledge, UFeed is the first system that fixes both the mediated schema and mappings based on user feedback over query answers. UFeed accepts as input a mediated schema and mappings between each source schema and this mediated schema. UFeed does not make any assumptions about the techniques used to create the initial mediated schema and mappings (the setup phase), and can work with any technique for schema matching and mapping. We demonstrate this by using two existing approaches from the literature: a holistic data integration approach that outputs one mediated schema for all data sources and one mapping for each source schema, and a probabilistic data integration approach that outputs several mediated schemas, each associated

CIKM'17 , November 6–10, 2017, Singapore, Singapore
© 2017 Association for Computing Machinery.
ACM ISBN 978-1-4503-4918-5/17/11…$15.00
https://doi.org/10.1145/3132847.3132887

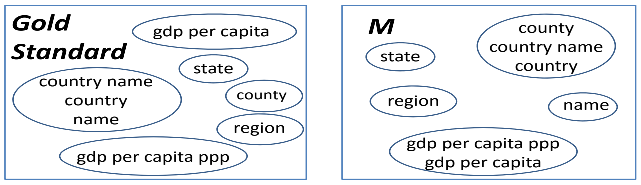

Gold Standard
gdp per capita
state
country name country name
county
region
gdp per capita ppp
M
county country name country
state
region
name
gdp per capita ppp gdp per capita

Figure 1: The gold standard G for integrating schemas S_1, S_2, S_3, and S_4 in Example 1, and a possible mediated schema M.

Mapping	Attributes Mapping

Map_1	county → {county, country name, country}
	state → {state}
	region → {region}

Map_1^G	county → {county}
	state → {state}
	country → {country, country name, name}
	region → {region}

Map_2	country name → {county, country name, country} gdp per capita → {gdp per capita ppp, gdp per capita} region → {region}

Map_2^G	country name → {country, country name, name}
	gdp per capita → {gdp per capita}
	region → {region}

Map_3	country → {county, country name, country} region → {region}
Map_3^G	country → {country, country name, name} region → {region}

Map_4	name → {name}
	gdp per capita ppp → {gdp per capita ppp, gdp per capita}
	region → {region}

Map_4^G	name → {country, country name, name} gdp per capita ppp → {gdp per capita ppp} region → {region}

Figure 2: The mappings to the mediated schema M and to the gold standard G.

with a probability, and several mappings for each source schema, each also associated with a probability (more details in Section 3).

Our philosophy in UFeed is that it is better to isolate the user from the details of the mediated schema and mappings, and not to assume that the user has background knowledge about all available data sources. Therefore, we do not require the user to issue her queries on the mediated schema. Instead, we let the user query an individual data source of her choice whose schema she is familiar with, similar to prior work [11]. UFeed maps the query on this data source to a query on the mediated schema and on other data sources. When query answers are shown to the user, she can mark any answer tuple as "correct" or "incorrect". The query itself and this feedback trigger refinement operations in UFeed.

The main contribution in UFeed is to propose a set of well-defined refinement operations that are triggered by the user's interactions with the system. These refinement operations modify the mediated schema and mappings with the goal of improving the quality of query answers. Modifying the mediated schema and mappings presents several challenges: Which source attributes should be part of the mediated schema? What causes an answer to be incorrect (error in the mediated schema and mappings or error in the data)? What if a feedback instance provided by the user is incorrect? UFeed addresses these challenges in its definition of the refinement operations and how they are triggered. Next, we present an example of the problems that UFeed needs to tackle in its refinement. We use this as a running example throughout the paper.

EXAMPLE 1. *Consider the following source schemas:*
S_1(county, state, country, region)
S_2(country name, gdp per capita, region)
S_3(country, region)
S_4(name, gdp per capita ppp, region)
Figure 1 shows the gold standard G for integrating the four data sources (i.e., the correct mediated schema, which we created manually), and a mediated schema M that can be the output of some automatic data integration approach. The mediated schema M differs from the gold standard in that it includes the attribute "county" in the same mediated attribute as {country name, country}, which is supposed to represent the concept "name of a country". The attribute "name" also represents the same concept but it is not part of the same mediated attribute. The mediated schema also combines "gdp per capita" and "gdp per capita ppp". The first represents nominal GDP while the second represents GDP at purchase power parity. These two concepts are related, but distinct, so they should not be part of the same mediated attribute. Figure 2 shows the mapping of each source schema to both the mediated schema and the gold standard. When a query is issued against the mediated schema M, the returned answers can have correct answers, missing answers, and/or incorrect answers. For example, when selecting states in a region, all answers are expected to be correct and complete since there are no mistakes in the "state" or "region" attributes in M. However, errors can occur when selecting countries in a region. For example, if a query is issued where the mediated attributes {name} and {region} are chosen, this query will return only a subset of the correct answers because no answers will be returned using "country" from S_1 and S_3, and "country name" from S_2. If the query uses the mediated attributes {county, country name, country} and {region}, it will return some correct answers based on the source attributes "country" and "country name", but it will also return incorrect answers based on the source attribute "county". Moreover, the answers from the data source S_4 with the attribute "name" will be missing.

The goal of UFeed is to address problems such as the ones presented in the previous example. UFeed observes queries and user feedback and refines the mediated schema and mappings until they are correct.

The contributions of this paper are as follows:

- Closing the loop of pay-as-you-go data integration by triggering the UFeed operations based on the user's actions in the process of querying available data sources.
- Defining a set of operations that refine automatically generated mediated schemas and mappings.
- Evaluating UFeed on real web data sources and showing that it improves the quality of query answers.

2 RELATED WORK

Data integration aims at automatically creating a unified mediated schema for a set of data sources and generating mappings between these data sources and the mediated schema. Schema matching and mapping have been extensively studied in the literature [6, 24], and the state of the art is that many matching decisions can be made automatically. Whenever ambiguity arises, involvement of an experienced user (e.g., a data architect) is required. To account for uncertainty faced by data integration systems due to this ambiguity, probabilistic models of data integration have emerged [11, 13, 20].

Involving users in various tasks related to data integration has been studied in the literature. The Corleone [18] system focuses on the entity matching problem and outsources the entire workflow to the crowd including blocking and matching. In [2], user feedback is used to choose the best data sources for a user in a relational data integration setting, and the best mediated schema for these sources. The system in [9] relies on writing manual rules to perform information extraction or information integration operations. The output view of these operations is then subject to feedback from the user in the form of inserting, editing, or deleting data. This feedback is then reflected on the original data sources and propagated to other views. In the Q system [28, 29, 31], keywords are used to match terms in tuples within the data tables. Foreign keys within the database are used to discover "join paths" through tables, and query results consist of different ways of combining the matched tuples. The queries are ranked according to a cost model and feedback over answers is used to adjust the weights on the edges of the graph to rerank the queries. Other approaches [4, 23] require the user to specify samples of data mapped from a source schema to a target schema in order to generate the mappings between these schemas.

In the context of schema matching and mappings, which is the focus of this paper, there has been work on involving users in the process to overcome the challenges of making all data integration decisions automatically. The following systems focus on directly changing the mappings prior to their use by end-users: In [7], the system provides functionalities for checking a set of mappings to choose the ones that represent better transformations from a source schema to a target schema. In [10], a debugger for understanding and exploring schema mappings is introduced. This debugger computes, and displays, on request, the relationships between source and target schemas. Muse [3] refines partially correct mappings generated by an automatic matcher, and asks the user to debug them by examining user-proposed examples. In [19], a large set of candidate matches can be generated using schema matching techniques. These matches need to be confirmed by the user. The candidate matches are sorted based on their importance (i.e., they are involved in more queries or associated with more data). A user is asked to confirm a match with a "yes" or "no" question. Similar work has also been proposed in [22, 30, 32]. The step of verifying schema mappings is done as part of setting up the data integration system. This results in a significant up-front cost [17]. In contrast to these approaches, UFeed promotes a pay-as-you-go approach, in which we use automatic techniques to create an initial mediated schema and generate semantic mappings to this schema. We then utilize user feedback to refine the mediated schema and mappings incrementally. UFeed infers the parts of the mediated schema and mappings that need to be modified by relying on user queries and feedback on query answers, while shielding the user from the details of the matching and mapping process. UFeed is also agnostic to the degree of confidence the data integration system has about its matches because user feedback is used to directly change the mediated schema and mappings, regardless of whether the data integration system is certain or uncertain about them.

Closest to our work is [5], where the focus is on refining alternative mappings in a pay-as-you-go fashion. The mediated schema is assumed to be correct, and the user issues a query along with constraints over the required precision and recall to limit the number of mappings used to answer the query. The user can give feedback over the returned answers so that the mappings can be annotated with an estimate of their precision and recall. These estimates are used in future queries to refine and select the mappings that would return the level of precision and recall desired by the user. UFeed refines not only the mappings, but also the mediated schema, without overburdening the user with specifying constraints on the quality of query answers. In this paper, as a comparison to [5], we show that refining the mappings alone is not sufficient to find high-quality answers to the user's queries.

3 PRELIMINARIES

3.1 Schema Matching and Mapping

Source Schema: In this paper, we focus on the integration of web tables, which usually have simple schemas that do not adhere to explicit data types or integrity constraints. Thus, a source schema consists of a set of attribute names.

Definition 1. *A source schema S that has n source attributes is defined by:* $S = \{a_1, \ldots, a_n\}$.

For q source schemas, the set of all source attributes in these schemas is $\mathcal{A} = attr(S_1) \cup \cdots \cup attr(S_q)$.

Mediated Attribute: A mediated attribute mA is a grouping of source attributes from different source schemas. Source attributes in a mediated attribute represent the same real-world concept.

Definition 2. *A mediated attribute is defined by:* $mA = \{S_i.a_x, \ldots, S_j.a_y | \forall i, j, i \neq j\}$.

Mediated Schema: In this paper, we require one mediated schema to be generated for a number of data sources belonging to the same domain. This approach is known as *holistic schema matching* [27], in contrast to approaches that perform pairwise matching between a pairs of schemas. Holistic schema matching is the most appropriate approach for web data integration because a large number of data sources needs to be covered by one mediated schema.

Definition 3. *A mediated schema $\mathcal{M}$ is defined by:* $\mathcal{M} = \{mA_1, \ldots, mA_m\}$*, where m is the number of mediated attributes in the mediated schema.*

Mapping: The mapping from any data source to the mediated schema is represented by a set of correspondences, each between a source attribute and a mediated attribute.

Definition 4. *A mapping $\mathcal{M}ap_i$ between source schema S_i and the mediated schema $\mathcal{M}$ is defined by:* $\mathcal{M}ap_i = \{a_j \rightarrow mA_k | j \in [1, |\mathcal{S}_i|], k \in [1, |\mathcal{M}|]\}$.

The process of generating a mediated schema holistically and generating mappings between each source schema and the mediated schema is referred to in this paper as *holistic data integration*.

3.2 Probabilistic Mediated Schemas and Mappings

The probabilistic model of data integration [11, 13] reflects the uncertainty faced by automatic approaches when integrating heterogeneous data sources. A probabilistic mediated schema can possibly consist of several mediated schemas, each of which is associated with a probability that reflects how likely the mediated schema represents the domain of the data sources.

Definition 5. *A probabilistic mediated schema $\mathcal{PM}$ with p mediated schemas is defined by: $\mathcal{PM} = \{(\mathcal{M}_1, Pr(\mathcal{M}_1)), \ldots, (\mathcal{M}_p, Pr(\mathcal{M}_p)))\}$, where $Pr(\mathcal{M}_i)$ is the probability that mediated schema $\mathcal{M}_i$ is the correct one.*

Similarly, a probabilistic mapping is defined between each source schema $\mathcal{S}_i$ and mediated schemas $\mathcal{M}_j$. The probabilistic mapping can possibly consist of several mappings each of which is associated with a probability.

Definition 6. *A probabilistic mapping $\mathcal{PM}ap_{ij}$ is defined by: $\mathcal{PM}ap_{ij} = \{(\mathcal{M}ap_1, Pr(\mathcal{M}ap_1)), \ldots, (\mathcal{M}ap_l, Pr(\mathcal{M}ap_l))\}$, where $l \geq 1$ is the number of mappings between source schema $\mathcal{S}_i$ and mediated schema $\mathcal{M}_j$.*

When queries are issued against the probabilistic mediated schema, answer tuples are computed from each possible mediated schema, and a probability is computed for each answer tuple. Details can be found in [11]. We refer to this model as *probabilistic data integration*, and we show that UFeed refinement applies to probabilistic data integration as well as holistic data integration.

3.3 Query Answering

In this paper, we focus on *select-project (SP)* queries using a SQL-like syntax. Supporting joins and more complex queries is left as future work. A query has a *SELECT* clause and a *WHERE* clause. There is no *FROM* clause because queries are issued over all data sources. This type of queries conforms with prior work [11].

As mentioned earlier, the user is completely isolated from the details of the mediated schema and mappings. Therefore, this approach is more suitable for users who are not experts and do not have detailed knowledge about the semantics of all queried data sources. The user writes a query over one source schema S_i. For example, a query over source schema S_3 in Example 1 is *SELECT country WHERE region = North America*. The system rewrites the query over the source schema to a query over the mediated schema. This is done by replacing each source attribute with the mediated attribute it maps to in $\mathcal{M}ap_3$. In our example query, the rewritten query becomes *SELECT {county, country name, country} WHERE {region} = North America*. If UFeed is not able to replace all source attributes in the query with mediated attributes, the query is only issued over data source S_3.

Once a query over the mediated schema is obtained, UFeed rewrites the query using the appropriate mappings so that it can be issued over all relevant data sources. For each source schema, if there is a source attribute that maps to a mediated attribute in the query, the query is rewritten so that the source attribute replaces the mediated attribute in the *SELECT* or *WHERE* clause. The query is rewritten for all data sources that are represented in the mediated attributes in the query. The rewritten queries are issued over the data sources and the answers are combined using a union operation.

4 REFINEMENT IN UFEED

In this section, we describe how UFeed accepts and stores user feedback and the operations triggered by this feedback. In using feedback to refine the mediated schema and mappings, UFeed has to address the following challenges: 1. Which source attributes should be in the mediated schema? Typically, data integration systems do not include all attributes from all data sources in the mediated schema. Doing so would make the mediated schema too large and semantically incoherent, and mappings too complex and difficult to use. Choosing source attributes based on their frequency in data sources has been used in prior work [11]. Whether this or some other method is used, the choice of attributes will not be perfect. Desired attributes may be excluded, and undesired ones may exist in the mediated schema. Even if a suitable frequency threshold is found for a specific domain, this threshold may be different for other domains. 2. What happens if UFeed receives conflicting feedback or performs incorrect refinement? One way to ensure correct feedback is to use feedback that is aggregated from multiple users over a period of time [12, 22]. However, even with this type of feedback aggregation, some of the feedback used by UFeed may be incorrect and result in incorrect refinements. UFeed needs to correct its mistakes based on future instances of correct feedback. 3. How should UFeed respond when the user marks a tuple in a query answer as incorrect? Is the answer incorrect because of an incorrect grouping of source attributes in a mediated attribute, an incorrect mapping from a source attribute to a mediated attribute, or because the data in the data source is incorrect? UFeed should pinpoint the origin of an error using only feedback over query answers. 4. How to adapt mappings to changes in the mediated schema? As the mediated schema is refined, some of the mappings are invalidated and some new ones need to be generated. UFeed should solve this problem without being dependent on the specific algorithm used to generate the mappings.

4.1 Attribute Correspondence and Answer Association

We first describe how UFeed represents user feedback. When a user issues a query and receives answer tuples, she can mark any of the answer tuples as "correct" (positive feedback) or "incorrect" (negative feedback). This is referred to as a *feedback instance*. Note that the user is not required to provide feedback on all answer tuples. She can choose as many or as few answer tuples as she wants to mark as "correct" or "incorrect".

Each feedback instance updates two in-memory data structures used by UFeed: *attribute correspondence set* and *answer association set*. An *attribute correspondence* links a source attributed used in the original query to a source attribute from another data source used in the rewritten query. This link means that the two source attributes represent the same concept. To illustrate using the query in Section 3.3, the original query uses the attribute *country* which is rewritten to *country name* when issuing the query over S_2. If feedback is received over an answer tuple based on this rewritten query, the attribute correspondence entry (*country, country name*) is created and associated with the type of feedback received (positive or negative). The attribute correspondence set stores all the attribute correspondences inferred from feedback that is received by UFeed.

An *answer association* links a value in an answer tuple to the source attribute in the rewritten query that this value comes from. For example, (*country name, "USA"*) is an answer association. The answer association set stores all the answer associations derived from user feedback.

The attribute correspondence set and answer association set capture all the feedback received by UFeed in a way that allows the system to refine the mediated schema and mapping based on this feedback.

4.2 UFeed Operations

UFeed has a set of abstract and independent operations that target multiple kinds of flaws in the mediated schema and mappings: adding/removing source attributes to/from the mediated schema, modifying mappings, and merging/splitting mediated attributes. Following, we describe these operations and how they are triggered.

4.2.1 Inject. The *Inject* operation overcomes the problem of missing source attributes in the mediated schema by adding source attributes that the user requires to the mediated schema. As discussed in Section 3, queries are formulated over one of the source schemas. Querying an attribute that exists in a source schema but not in the mediated schema is a sufficient indication that this attribute is important to the user and needs to be injected in the mediated schema. This triggers the *Inject* operation. The main question for *Inject* is which mediated attribute the new source attribute should join. UFeed uses a minimum distance classifier to answer this question. The minimum distance classifier chooses the mediated attribute that has the source attribute that is most similar to the newly added source attribute (i.e., nearest neighbor). Other types of classifiers can also be used [14]. A threshold α is introduced so that a source attribute is not forced to join a mediated attribute to which it has a relatively low similarity. We use a value $\alpha = 0.8$. If the new source attribute cannot join any existing mediated attribute, it is placed in a new mediated attribute that contains only this source attribute. Thus, *Inject* can be defined as follows:

Definition 7. *If the current set of source attributes in the mediated schema is $\mathcal{A}' = attr'(S_1) \cup attr'(S_2) \cup \ldots \cup attr'(S_q)$, where $attr'(S_i)$ is the set of source attributes of data source S_i that contribute to the mediated schema. Inject($S_i.a$) performs two steps: 1. $\mathcal{A}' \leftarrow \mathcal{A}' \cup \{S_i.a\}$, and 2. $mA_i \leftarrow mA_i \cup \{S_i.a\}$ for mA_i with the highest similarity to $S_{i.a}$ greater than α OR $mA_{|M|+1} \leftarrow \{S_i.a\}$ if no mA_i has similarity to $S_i.a$ greater than α.*

4.2.2 Confirm. This operation is triggered when the user marks an answer tuple "correct". Since the answer tuple is correct, this means that the data, the mediated attributes, and mappings used to generate this tuple are correct. The correctness of the data is recorded in the answer association set, and the correctness of the mappings is recorded in the attribute correspondence set. In UFeed, there are two kinds of confirmations: *definite confirmations* and *tentative confirmation.* A definite confirmation is applied to the attribute correspondences and answer associations that are directly touched by the user feedback instance. A tentative confirmation is applied to the answer associations that are indirectly touched by this feedback instance. For example, consider the query *SELECT country WHERE region = North America* over S_3. This query is rewritten based on Map_1 to be issued over S_1. The rewritten query is *SELECT county WHERE region = North America.* The same query is also rewritten based on Map_2 to be issued over S_2. The rewritten query is *SELECT country name WHERE region = North America.* The answers are shown in Figure 3. Giving positive feedback over the answer *"Canada"* leads to the creation of the attribute correspondences (S_3*.country*, S_2*.country name*) and (S_3*.region*, S_2*.region*), and the answer associations (S_2*.country name*, *"Canada"*) and (S_2*.region*, *"North America"*). Notice that there are also answer associations for the source attributes from S_3, but we omit them for the sake of space and clarity of the example. The aforementioned confirmations are all definite, since they are based directly on the tuple "Canada" over which the user provided positive feedback. Definite confirmations are represented in the figure by a solid green line. Other answer tuples that are generated by the same rewritten query are given tentative confirmations, represented by a dotted green line in the figure. Assigning tentative confirmations is based on the reasoning that the positive feedback provided by the user indicates that the value of the source attribute on which this feedback was given is correct, and this source attribute is indeed an instance of the mediated attribute. Other values of the source attribute are likely to be correct, so they should be confirmed. However, there may be errors in the data resulting in some of these values being incorrect. Therefore, the confirmation remains a tentative confirmation, and the confirmation of a value becomes definite only if the user explicitly provides positive feedback on this value.

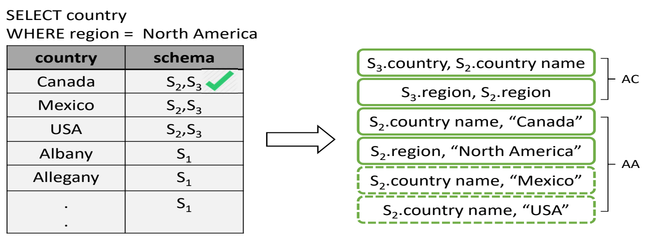

SELECT country
WHERE region = North America
country
schema
Canada
S2,S3
Mexico
S2,S3
USA
S2,S3
Albany
S1
Allegany
S1
S1
S3.country, S2.country name
S3.region, S2.region
S2.country name, "Canada"
S2.region, "North America"
S2.country name, "Mexico"
S2.country name, "USA"
AC
AA

Figure 3: Positive feedback over an answer tuple and the resulting attribute correspondences (AC) and answer associations (AA).

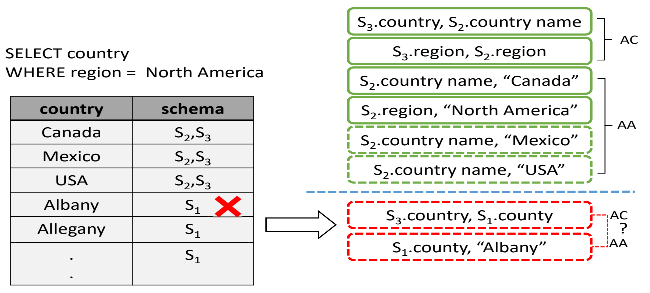


Figure 4: Negative feedback over an answer tuple and the resulting linking of the attribute correspondence (AC) and answer association (AA) to which the feedback applies. Attribute correspondences and answer associations from the previous query are shown above the blue dotted line.

The *Confirm* operation aims at protecting source attributes in the mediated attributes from being affected by other operations that alter the mediated schema, in particular, the *Split* and *Blacklist* operations that will be discussed next.

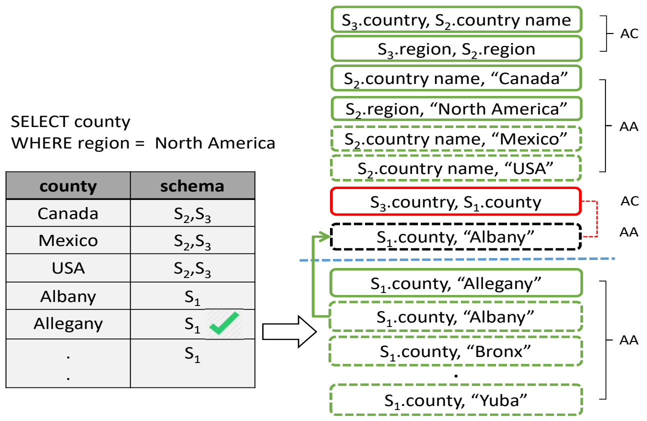


Figure 5: Positive feedback over an answer tuple to a query asking for counties in North America.

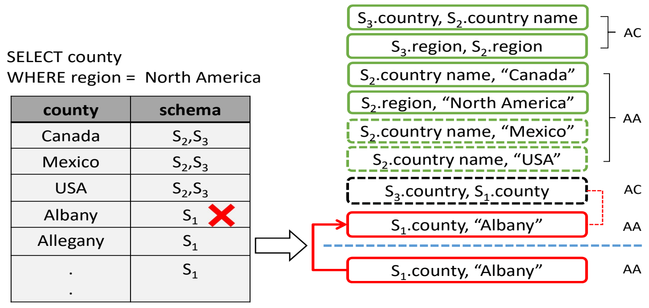


Figure 6: Negative feedback over an answer tuple to a query asking for counties in North America.

4.2.3 Split and Blacklist. When negative feedback is received over an answer tuple, this means that either the data is incorrect or the mediated schema and mappings are incorrect. In particular, one or more attribute values in the source tuple may be incorrect, or the source attribute does not represent the same concept as the mediated attribute it is part of. UFeed reflects this information on the attribute correspondences and answer associations created for this feedback instance. The attribute correspondences and answer associations for this feedback instance are *linked* together for future investigation based on future feedback. Figure 4 shows the uncertainty faced by UFeed when negative feedback is received over the answer *"Albany"*. UFeed does not know if the answer is incorrect because *country* and *county* should not be in the same mediated attribute, or because *"Albany"* is not a *county*. The attribute correspondence and answer association are linked together as shown in the figure (represented by a dotted red line).

Now, consider the query: *SELECT county WHERE region = North America* and its answers in Figure 5. Assume that positive feedback is received over *"Allegany"*. As explained earlier, a definite confirmation is applied to (S_1*.county, "Allegany"*). Note that no attribute correspondence is added because this answer tuple comes from the source schema over which the query is issued. Tentative confirmations are applied to the remaining answer associations as explained earlier. However, the answer association (S_1*.county, "Albany"*) has been previously linked to a negative feedback instance. Conflicting feedback, as we will explain later in this section, results in updating the status of the entry to "unknown" (represented by black dotted line), that is, neither correct nor incorrect. With this update, UFeed concludes that the reason for the negative feedback received in Figure 4 is that *county* should not be in the same mediated attribute as *country* (because *county* is the source attribute used to generate the answer "Albany"). This triggers the *Split* operation, which splits the source attribute used in the rewritten query from the mediated attribute it is part of, and forms a new mediated attribute that only contains this one source attribute. In this example, *county* is removed from the mediated attribute {*county, country name, country*} and the new mediated attribute {*county*} is added to the mediated schema. If the split source attribute is the only member of a mediated attribute, the mediated attribute is also removed. If the split source attribute does not exist in the mediated attribute, the correspondence between the source attribute and the mediated attribute in the mapping used to answer the query is removed. Thus, the *Split* operation is defined as:

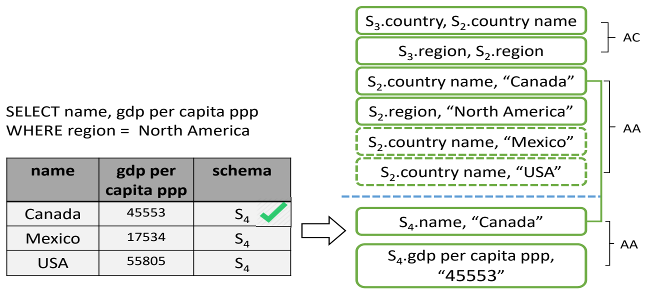
Definition 8. *If $S_i.a$ is a source attribute, where $S_i.a \in \mathcal{A}'$, $Split(S_i.a, mA_x)$ performs three possible actions:*

$$\begin{cases} Remove(mA_x), \mathcal{A}' \leftarrow \mathcal{A}' - \{S_i.a\} \\ \qquad if\, S_i.a \in mA_x \text{ } AND \text{ } |mA_x| = 1 \\ OR\ 1.\ mA_x \leftarrow mA_x - S_i.a \quad 2.\ mA_{|\mathcal{M}|+1} \leftarrow \{S_i.a\} \\ \qquad if\, S_i.a \in mA_x \text{ } AND \text{ } |mA_x| > 1 \\ OR\ Remove(S_i.a \rightarrow mA_x) \\ \qquad if\, S_i.a \notin mA_x \end{cases}$$

UFeed uses the following heuristic: When the user provides negative feedback indicating that an answer tuple is incorrect, UFeed assumes that there is only one mistake that caused this answer tuple to be incorrect. This can be a mistake in the mediated schema or mappings, or it can be erroneous data. If it happens that multiple mistakes cause an answer tuple to be incorrect, UFeed will fix the mistakes one by one based on multiple instances of negative feedback.

To illustrate another way UFeed identifies the cause of negative feedback, assume that instead of giving positive feedback over *"Allegany"*, the user provides negative feedback over *"Albany"*, as shown in Figure 6. This feedback is incorrect since *"Albany"* is in fact a county, but it serves our example. In this case, UFeed does not face uncertainty about the reason for this negative feedback because this answer tuple is generated from one source (S_1), without using any mappings. UFeed knows now that *"Albany"* is erroneous data in the data source. This triggers the *Blacklist* operation. This operation maintains a blacklist that keeps track of incorrect answer associations in the answer association set. The blacklist is used in the query answering process to remove erroneous data from query answers. In our example, the blacklist removes *"Albany"* from future answers. This negative feedback also updates the status of the attribute correspondence (S_3*.country*, S_1*.county*) to "unknown" until future feedback indicates it is incorrect.

4.2.4 Merge. This operation is triggered when two or more answer associations share the data value while having different source attributes. For example, consider the query in Figure 7, which finds countries and their "gdp per capita purchase power parity" values in North America. Assume this query is issued after the



```
SELECT name, gdp per capita ppp
WHERE region =  North America
```

name	gdp per capita ppp	schema
Canada	45553	S_4 ✓
Mexico	17534	S_4
USA	55805	S_4

S_3.country, S_2.country name
S_3.region, S_2.region
AC
S_2.country name, "Canada"
S_2.region, "North America"
S_2.country name, "Mexico"
S_2.country name, "USA"
AA
S_4.name, "Canada"
S_4.gdp per capita ppp, "45553"
AA

Figure 7: Positive feedback that results in triggering the *Merge* operation.

query in Figure 3. Receiving positive feedback over the answer tuple (*"Canada", "45553"*) applies a definite confirmation to the answer associations (S_4*.name, "Canada"*) and (S_4*.gdp per capita ppp, "45553"*). However, the answer association (S_2*.country name, "Canada"*) exists and was confirmed by the user. This triggers the *Merge* operation, which merges the two mediated attributes that the two source attributes in the answer associations are in. The *Merge* operation is triggered when two answer associations that share the data value are confirmed with definite or tentative confirmation.

Definition 9. *If $S_i.a$ and $S_j.b$ are two source attributes, where $S_i.a, S_j.b \in \mathcal{A}'$, $Merge(S_i.a, S_j.b)$ performs two steps: 1. $mA_i \leftarrow mA_i \cup mA_j | S_i.a \in mA_i, S_j.b \in mA_j$. 2. $Remove(mA_j)$.*

4.2.5 Adapt. This operation updates the mappings whenever the mediated schema is changed by other UFeed operations. It is essential for maintaining semantic coherence as the mediated schema changes. Formally, when a mediated schema $\mathcal{M}$ is changed into a new mediated schema $\mathcal{M}'$ via any of the aforementioned UFeed operations, the mapping $\mathcal{M}ap_i$ for any source schema S_i that is affected by the changes should evolve into $\mathcal{M}ap'_i$ that maps S_i to $\mathcal{M}'$.

The UFeed operations that trigger the *Adapt* operation are: *Inject*, *Split*, and *Merge* because these are the operations that make changes to the set of attributes in the mediated schema $\mathcal{A}$ and their groupings into mediated attributes.

Mapping adaptation after *Inject*: The *Inject* operation introduces a new source attribute to the set of source attributes in the mediated schema. This change in the mediated schema will require updating the mapping of the source schema that the injected source attribute is part of. Mapping adaptation in this case is straightforward. Following Definition 7, if the injected source attribute is $S_i.a_x$ the new mapping becomes $\mathcal{M}ap'_i = \mathcal{M}ap_i \cup (S_i.a_x \rightarrow mA_j)$, where $S_i.a_x \in mA_j \wedge (!\exists S_i.a_y \in mA_j | x \neq y)$. This means that the injected source attribute should map to the mediated attribute it is injected in, unless another source attribute from the same source schema is in this mediated attribute.

Mapping adaptation after *Split*: Definition 8 has three different cases for the *Split* operation. Different mapping adaptations apply in each case.

$$\begin{cases} Remove(S_i.a \rightarrow mA_x) \\ OR\ S_i.a \rightarrow mA_{|M|+1} \\ OR\ \text{No Action} \end{cases}$$

For the first case, where the source attribute and the mediated attribute are removed, the correspondence to the mediated attribute is also removed. For the second case, where one source attribute is split from the mediated attribute, the split source attribute maps to the new mediated attribute that only contains the split source attribute. The third case removes the mapping, so no adaptation is needed.

Mapping adaptation after *Merge*: When two mediated attributes are merged according to Definition 9, the mappings from any source attribute to any of the merged mediated attributes should be updated. Following Definition 9, mappings to the mediated attribute mA_i do not need to be updated. Only mappings to the mediated attribute mA_j should be changed to map to mA_i.

4.3 Applying UFeed Operations to Probabilistic Mediated Schemas and Mappings

As noted earlier, UFeed operations can be applied on probabilistic mediated schemas and mappings. The challenge in this case is that there are several mediated schemas, each with an associated probability. UFeed uses a simple solution to address this challenge: it deals with each mediated schema independently, as if it were the output of a holistic schema matching system. Whenever two mediated schemas are equivalent (have the exact same grouping of source attributes), one is removed. The ultimate goal of schema refinement in this case is for the probabilistic mediated schema to converge to a single (non-probabilistic) mediated schema.

Calculating probabilities for mediated schemas and mappings in UFeed is dependent on the details of the probabilistic data integration approach. UFeed should incorporate the probability calculation method in order to update the probabilities as the probabilistic mediated schema and mappings are refined. In this paper, we incorporated the probability calculation method of [11].

4.4 Handling Incorrect Feedback

UFeed expects incorrect feedback to be rare, but it can cancel its effects through future correct feedback instances. UFeed does not know that a feedback instance is incorrect, but makes changes independently based on one feedback instance at a time. Therefore, more correct feedback instances will ultimately undo the negative effects caused by incorrect feedback.

Inject: An incorrect *Inject* can be fixed using the *Split* operation. If the injected source attribute is incorrectly placed in an existing mediated attribute, negative feedback over answers that are generated using the injected source attribute will split it from the mediated attribute. If the source attribute forms a mediated attribute on its own, negative feedback removes this source attribute, and consequently the mediated attribute, from the mediated schema.

Confirm: When an attribute correspondence is mistakenly given a definite confirmation, the source attribute cannot be split from the mediated attribute. This confirmation cannot be removed unless a negative feedback is received and it is known that the error comes from the attribute correspondence and not an answer association. In this case, the confirmation is removed and the status of the attribute correspondence is updated to "unknown".

Split: If a source attribute is mistakenly split due to incorrect feedback, this can be simply undone by receiving correct feedback that triggers the *Merge* operation, which merges the source attribute that was split with the mediated attribute it was split from. If the *Split* results in removing the source attribute from the mediated schema, this will require the user to query the removed source attribute to trigger an *Inject* operation to add it to the mediated schema.

Blacklist: To avoid blacklisting a value that may be correct due to incorrect feedback, we implement a *second chance* technique for this specific operation. The second chance technique allows the value to be shown to the user after it is identified by UFeed as erroneous. It is allowed to be shown to the user until another negative feedback is received over it. This value is then blacklisted in future queries.

Merge: An incorrect merge operation that merges two mediated attributes can be undone by receiving negative feedback over any of the merged source attributes. This will trigger a *Split* operation that will split the mistakenly merged source attribute from the new mediated attribute.

5 EXPERIMENTAL EVALUATION

5.1 Experimental Setup

We ran our experiments on a machine with Intel Core i7 CPU at 2.6 GHz and 8 GB of memory. To generate the starting mediated schema and mappings for UFeed, we use the techniques in [25] for the holistic data integration approach and the techniques in [11] for the probabilistic data integration approach. These two techniques are the state-of-the-art in holistic and probabilistic data integration.

Data Sources: We extracted our data sources from Google Fusion Tables [1]. To find a set of data sources representing a given domain, we issue a keyword search representing that domain on Google Fusion Tables. We consider the top 150 tables returned and manually refine them to eliminate redundant or irrelevant ones. We feed relevant tables to our data integration algorithms (holistic and probabilistic) to create the mediated schema(s) and mappings. The mediated schema(s) and mappings are used to answer queries formulated over any data source by returning answers from other data sources that can contribute to the answer set, as described in Section 3. Table 1 presents the data (domains and tables) that we used in our experiments. The holistic data integration approach generates one mediated schema and one set of mappings for each domain. The probabilistic data integration approach generates one mediated schema for the "Movies" domain, and two mediated schemas for each of the "World GDP" and "Internet Usage" domains. The number of probabilistic mappings for each source in the three domains ranged from 0 mappings, where the source does not map to the mediated schema to 8 different possible mappings from one source to the mediated schema of the domain.

Gold Standard: For each domain, we manually create a gold standard mediated schema and the corresponding mapping for each data source. When used, the gold standard returns correct and complete answers to all queries.

Queries: For each domain, we manually construct a set of queries that trigger the UFeed operations. These queries focus on the parts of the mediated schema that differ from the gold standard. When the queries are issued before UFeed refines the schema, they can return incomplete, incorrect, or empty answers. As the UFeed operations are applied, the quality of the query answers improves. We focus on these queries because queries that target parts of the mediated schema that are already the same as the gold standard will always return complete and correct answers, and thus offer limited opportunities for testing UFeed. The number of queries in each setting is shown in Table 2.

5.2 Quality of Query Answers

In this section, we evaluate whether UFeed improves the quality of query answers. To measure the quality of the current mediated schema (or schemas in the case of probabilistic data integration) and mappings, we run our queries over both the gold standard and the current mediated schema and mappings. Assuming the answer set from the gold standard is G and the answer set from the current mediated schema and mappings used by UFeed is A, we measure the quality of query answers using: *Precision* $P = \frac{|A \cap G|}{|A|}$, *Recall* $R = \frac{|A \cap G|}{|G|}$, and *F-measure* $F = \frac{2PR}{P+R}$. We compute the average precision, recall, and F-measure of the answers to all queries in the set of evaluation queries for each domain. In this paper, we report only the F-measure due to the lack of space, but we note that there is no large gap between precision and recall curve, so reporting them will not give additional insights.

In our experiments, we provide feedback to UFeed after each query, and UFeed immediately refines the mediated schema and mappings based on this feedback so that subsequent queries get higher quality answers. We generate feedback by comparing sample answers from the two answer sets A and G (defined above). However, we do not generate a feedback instance for each tuple in A and G. We randomly choose one tuple from A and look for it in G. If it exists, a positive feedback instance is generated. If it does not, a negative feedback instance is generated. After every UFeed operation, we rerun all queries in the evaluation set that were affected by this operation and recompute the F-measure.

The order of the queries run, and consequently the UFeed operations, has some effect on the quality of the query answers. To take into account the effect of query ordering, we repeat each experiment 10 times with a different random ordering of the queries and report the average F-measure for the 10 runs along with its 95% confidence interval. The random orderings are created under some simple ordering constraints. For example, a source attribute cannot be merged with another source attribute before it is injected in the mediated schema.

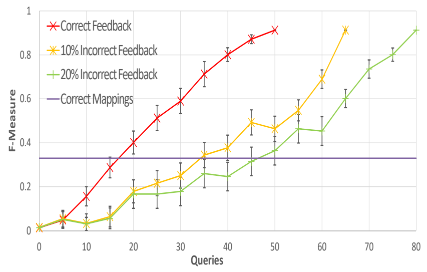
UFeed recovers from incorrect feedback by relying on future correct feedback. To evaluate the effectiveness of this approach, we test UFeed with 10% and 20% incorrect feedback instances chosen at random and show that UFeed can overcome their effect through additional correct feedback.

As mentioned earlier, to the best of our knowledge, the only other technique that uses feedback over query answer tuples is [5]. In that paper, query feedback is used to refine the mappings but not the mediated schema. To demonstrate that this is not sufficient, we generate correct mappings to the automatically generated mediated schema by manually modifying the results of the automatic mapping. We show the F-measure obtained using these correct mappings, which is the best that a technique that only refines the mappings such as [5] can obtain.

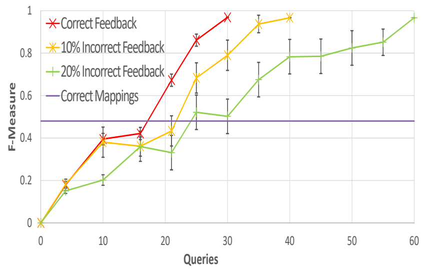
Figure 8 shows the quality of the query answers (F-measure) for the three domains shown in Table 1 for holistic and probabilistic

Keywords	Tables	Attributes/Table	Total Attributes	Tuples/Table	Total Tuples
Internet Usage	38	2–230	886	7–261	3826
Movies	20	1–16	117	10–3201	7422
World GDP	12	2–230	839	12–262	4830

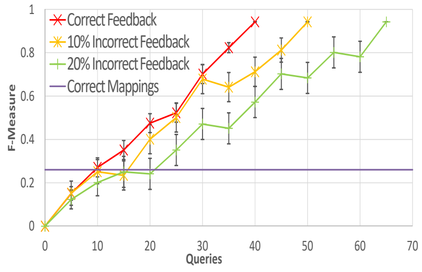
Table 1: Data sets used in the experiments.



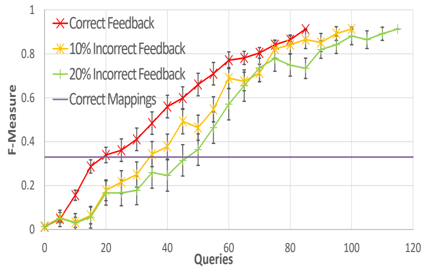
(a) Internet - Holistic



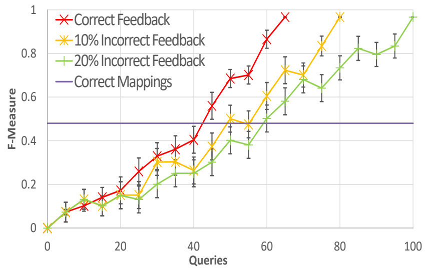
(b) Movies - Holistic



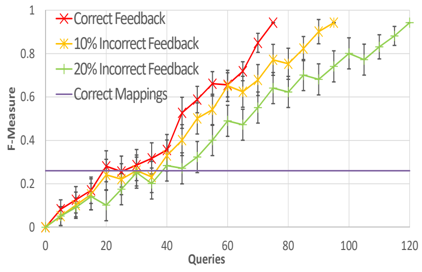
(c) World GDP - Holistic



(d) Internet - Probabilistic



(e) Movies - Probabilistic



(f) World GDP - Probabilistic

Figure 8: Quality of query answers.

	Internet Usage	Movies	World GDP
Holistic	49	32	40
Probabilistic	85	65	74

Table 2: Number of queries in the different settings.

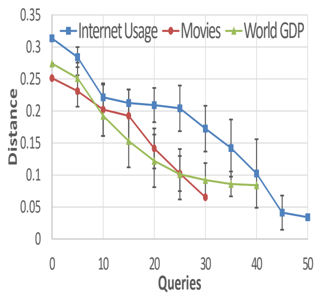
data integration. Since all the queries in the set of evaluation queries touch parts of the initial mediated schema that differ from the gold standard, the F-measure starts at a very low value. The figure shows that UFeed consistently reaches a high value of F-measure (> 0.9). As expected, UFeed is slower to converge when there is incorrect feedback, but the important point to note is that UFeed converges to the same (high) value of the F-measure even if up to 20% of the feedback is incorrect. The confidence intervals are narrow, showing that while the order of queries has some effect on the quality of the results, this effect is small.

The figure shows that having correct mappings (the best for a technique such as [5]) is not sufficient to guarantee high quality answers (the horizontal line in the plots). The automatically generated mappings can be refined to map to the correct mediated attribute, if it exists. However, there are typically mediated attributes that are either a group of source attributes representing different concepts (affecting precision), or an incomplete group that is missing relevant source attributes (affecting recall).

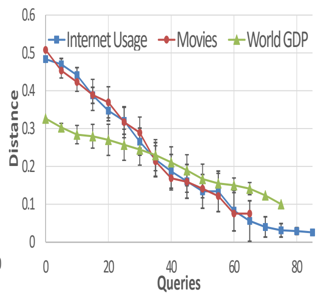
It is important to note that the user does not need to wait until the mediated schema is the same as the gold standard to receive benefit from UFeed. Whenever a UFeed operation is triggered, the answers to the query that triggered this operation are improved. Thus, there is immediate improvement to the answers that the user is currently interested in. When there is incorrect feedback, UFeed requires more operations (i.e., more queries) to converge, but still reaches a high value of F-measure.

5.3 Distance to the Gold Standard

Another question we ask is how close the current mediated schema is to the gold standard mediated schema. Since mediated schema generation can be viewed as a clustering problem (clustering source attributes into mediated attributes), we use the F-measure of clustering output [26] to measure the distance between any mediated schema and the gold standard. This measure is computed as follows: Each mediated attribute represents a *cluster* of source attributes. A *contingency matrix* is constructed for all the clusters in the gold standard and the current mediated schema. Each cell (c_G, c_M) in the matrix counts the number of source attributes that exist in the two clusters, c_G in the gold standard and c_M in the current mediated



(a) Holistic



(b) Probabilistic

Figure 9: Distance to the gold standard.

schema. The distance between the current mediated schema and the gold standard is computed as follows:

1. Precision, recall, and F-measure are computed for each cell (c_G, c_M), where $Precision(c_G, c_M) = \frac{|(c_G, c_M)|}{|c_M|}$, $Recall(c_G, c_M) = \frac{|(c_G, c_M)|}{|c_G|}$, and $F(c_G, c_M) = \frac{2PR}{P+R}$.

2. The F-measure of each gold standard cluster is computed as the maximum value of all the F-measures for this cluster: $F(c_G) = \max_{|c_M|}(F(c_G, c_M))$.

3. The F-measure of the clustering output is computed as the weighted average of the F-measures of all clusters of the gold standard: $F = \frac{\sum_{c_G \in C} |c_G| F(c_G)}{\sum_{c_G \in C} |c_G|}$.

4. Finally, the distance between the two mediated schemas is computed as $Distance = 1 - F$.

Figure 9 shows the distance between the current mediated schema after each query and the gold standard. The distance is shown for the three domains in the case where only correct feedback is used. For the probabilistic mediated schema, the mediated schema with the minimum distance to the gold standard is used. The figure shows that the distance steadily decreases as queries are issued and the UFeed operations are applied, which demonstrates the effectiveness and efficiency of UFeed.

6 CONCLUSION AND FUTURE WORK

We presented UFeed, a system that closes the loop of pay-as-you go relational data integration on the web by providing mechanisms to improve the quality of the mediated schema and mappings based on feedback received from the user. A key feature of UFeed is that it refines both the mediated schema and the mappings from the data sources to this schema. Another key feature is that the refinement relies on the queries issued by the user and feedback on the answers to these queries, without the need to directly manipulate the mediated schema or mappings. UFeed refinement is based on a set of well-defined operators, and it works for holistic and probabilistic data integration.

As future work, it is possible to extend UFeed to support more complex schemas and queries. UFeed currently supports selection queries on single-table schemas, and this covers a large fraction of the use cases on the web. Nevertheless, it would be useful to extend UFeed to support data sources and mediated schemas that consist of multiple tables. This requires extending UFeed to support queries with joins, issuing these queries to the data sources and learning from feedback on their answers.

REFERENCES

[1] Google Fusion Tables. http://research.google.com/tables.
[2] Ashraf Aboulnaga and Kareem El Gebaly. 2007. μbe: User guided source selection and schema mediation for internet scale data integration. In *ICDE*.
[3] Bogdan Alexe, Laura Chiticariu, Renée J Miller, and Wang-Chiew Tan. 2008. Muse: Mapping understanding and design by example. In *ICDE*.
[4] Bogdan Alexe, Balder ten Cate, Phokion G. Kolaitis, and Wang-Chiew Tan. 2011. Designing and refining schema mappings via data examples. In *SIGMOD*.
[5] Khalid Belhajjame, Norman W Paton, Suzanne M Embury, Alvaro AA Fernandes, and Cornelia Hedeler. 2013. Incrementally improving dataspaces based on user feedback. *Information Systems* 38, 5 (2013).
[6] Philip A. Bernstein, Jayant Madhavan, and Erhard Rahm. 2011. Generic schema matching, ten years later. In *PVLDB*.
[7] Angela Bonifati, Giansalvatore Mecca, Alessandro Pappalardo, Salvatore Raunich, and Gianvito Summa. 2008. Schema mapping verification: The spicy way. In *EDBT*.
[8] Michael J Cafarella, Alon Halevy, Daisy Zhe Wang, Eugene Wu, and Yang Zhang. 2008. WebTables: exploring the power of tables on the web. In *PVLDB*.
[9] Xiaoyong Chai, Ba-Quy Vuong, AnHai Doan, and Jeffrey F. Naughton. 2009. Efficiently incorporating user feedback into information extraction and integration programs. In *SIGMOD*.
[10] Laura Chiticariu and Wang-Chiew Tan. 2006. Debugging schema mappings with routes. In *VLDB*.
[11] Anish Das Sarma, Xin Dong, and Alon Halevy. 2008. Bootstrapping pay-as-you-go data integration systems. In *SIGMOD*.
[12] AnHai Doan and Robert McCann. 2003. Building data integration systems: A mass collaboration approach. In *Proc. of the Workshop on Information Integration on the Web*.
[13] Xin Dong, Alon Y. Halevy, and Cong Yu. 2007. Data integration with uncertainty. In *PVLDB*.
[14] Richard O. Duda, David G. Stork, and Peter E. Hart. 2000. *Pattern classification and scene analysis* (2nd ed.). Wiley.
[15] Julian Eberius, Maik Thiele, Katrin Braunschweig, and Wolfgang Lehner. 2015. Top-k entity augmentation using consistent set covering. In *SSDBM*.
[16] Hazem Elmeleegy, Jayant Madhavan, and Alon Halevy. 2009. Harvesting relational tables from lists on the web. In *PVLDB*.
[17] Michael Franklin, Alon Halevy, and David Maier. 2005. From databases to dataspaces: a new abstraction for information management. *SIGMOD Rec.* (2005).
[18] Chaitanya Gokhale, Sanjib Das, AnHai Doan, Jeffrey F. Naughton, Narasimhan Rampalli, Jude Shavlik, and Xiaojin Zhu. 2014. Corleone: Hands-off crowdsourcing for entity matching. In *SIGMOD*.
[19] Shawn R. Jeffery, Michael J. Franklin, and Alon Y. Halevy. 2008. Pay-as-you-go user feedback for dataspace systems. In *SIGMOD*.
[20] Matteo Magnani, Nikos Rizopoulos, Peter Mc. Brien, and Danilo Montesi. 2005. Schema integration based on uncertain semantic mappings. In *ER*.
[21] Hatem A. Mahmoud and Ashraf Aboulnaga. 2010. Schema clustering and retrieval for multi-domain pay-as-you-go data integration systems. In *SIGMOD*.
[22] Robert McCann, Warren Shen, and AnHai Doan. 2008. Matching schemas in online communities: A web 2.0 approach. In *ICDE*.
[23] Li Qian, Michael J. Cafarella, and H. V. Jagadish. 2012. Sample-driven schema mapping. In *SIGMOD*.
[24] Erhard Rahm and Philip A. Bernstein. 2001. A survey of approaches to automatic schema matching. *VLDB Journal* 10, 4 (2001).
[25] Len Seligman, Peter Mork, Alon Halevy, Ken Smith, Michael J. Carey, Kuang Chen, Chris Wolf, Jayant Madhavan, Akshay Kannan, and Doug Burdick. 2010. OpenII: an open source information integration toolkit. In *SIGMOD*.
[26] Michael Steinbach, George Karypis, and Vipin Kumar. 2000. A comparison of document clustering techniques. In *Proc. KDD Workshop on Text Mining*.
[27] Weifeng Su, Jiying Wang, and Frederick Lochovsky. 2006. Holistic schema matching for web query interfaces. In *EDBT*.
[28] Partha Pratim Talukdar, Zachary G. Ives, and Fernando Pereira. 2010. Automatically incorporating new sources in keyword search-based data integration. In *SIGMOD*.
[29] Partha Pratim Talukdar, Marie Jacob, Muhammad Salman Mehmood, Koby Crammer, Zachary G. Ives, Fernando Pereira, and Sudipto Guha. 2008. Learning to create data-integrating queries. In *PVLDB*.
[30] Steven Euijong Whang, Peter Lofgren, and Hector Garcia-Molina. 2013. Question selection for crowd entity resolution. In *PVLDB*.
[31] Zhepeng Yan, Nan Zheng, Zachary G. Ives, Partha Pratim Talukdar, and Cong Yu. 2013. Actively soliciting feedback for query answers in keyword search-based data integration. In *PVLDB*.
[32] Chen Jason Zhang, Lei Chen, H.V. Jagadish, and Chen Caleb Cao. 2013. Reducing uncertainty of schema matching via crowdsourcing. In *PVLDB*.