Pay-as-you-go Reconciliation in Schema Matching Networks

Nguyen Quoc Viet Hung[*], Nguyen Thanh Tam[*], Zoltán Miklós[†], Karl Aberer[*], Avigdor Gal[#], Matthias Weidlich[+]

[*] *École Polytechnique Fédérale de Lausanne,* [†] *Université de Rennes 1,* [#] *Technion - Israel Institute of Technology,* [+] *Imperial College London*

***Abstract*—Schema matching is the process of establishing correspondences between the attributes of database schemas for data integration purposes. Although several automatic schema matching tools have been developed, their results are often incomplete or erroneous. To obtain a correct set of correspondences, a human expert is usually required to validate the generated correspondences. We analyze this reconciliation process in a setting where a number of schemas needs to be matched, in the presence of consistency expectations about the network of attribute correspondences. We develop a probabilistic model that helps to identify the most uncertain correspondences, thus allowing us to guide the expert's work and collect his input about the most problematic cases. As the availability of such experts is often limited, we develop techniques that can construct a set of good quality correspondences with a high probability, even if the expert does not validate all the necessary correspondences. We demonstrate the efficiency of our techniques through extensive experimentation using real-world datasets.**

***Keywords*—data integration, schema matching, reconciliation, user feedback, information gain**

I. Introduction

More and more online services enable users to upload and share structured data, including Google Fusion Tables [24], Freebase [9], and Factual [2]. These services primarily offer easy visualization of uploaded data as well as tools to embed the visualisation to blogs or Web pages. As the number of publicly available datasets grows rapidly and fragmentation of data in different sources is a common phenomenon, it is essential to create the interlinks between them [11]. An example is the often quoted coffee consumption data found in Google Fusion Tables, which is distributed among different tables that represent a specific region [24]. Extraction of information over all regions requires means to query and aggregate across multiple tables, thereby raising the challenge of interconnecting schemas to achieve an integrated view of the data.

There is a large body of work on schema matching techniques; numerous commercial and academic schema matching tools, called matchers, have been developed in recent years [7], [36]. Even though matchers achieve impressive performance on some datasets, they cannot be expected to yield a correct result in the general case. Since matchers rely on heuristic techniques, their result is inherently uncertain. In practice, data integration tasks often include a post-matching phase, in which correspondences are reviewed and validated by an expert.

We focus on a setting in which matching is conducted for a network of related schemas that are matched against each other. Having a network of multiple schemas enables the introduction of network-level integrity constraints, thereby providing evidence for the quality of the matching by detecting constraint violations. Such integrity constraints are essential to enforce the natural expectations in the data integration applications. We encapsulate all of this information in a unified model, namely a *probabilistic matching network*, in which each attribute correspondence is associated with a probability. The notion of probabilistic matching networks is interesting in its own right and is beneficial in other scenarios, including enterprise applications [42] and mashup contexts [12]. In principle, a matching network enables collaborative integration scenarios, where an approach with a monolithic, mediated schema is too costly or simply infeasible.

In this paper, we go beyond the common practice of human reconciliation in improving and validating matchings for a pair of schemas. Instead, we study the reconciliation for a probabilistic matching network, in which the participating expert should respect the network-level integrity constraints to guarantee the overall matching quality. The presence of such integrity constraints creates a number of dependencies between correspondences, which may be hard to overlook especially in large-scale networks. Hence, reconciliation is a complex task. Despite this challenge, dependencies between correspondences open an opportunity to guide the expert's work by defining the order in which the expert gives his input (e.g. in which order to assert whether a correspondence is correct).

Guiding the selection of correspondences is essential for effective reconciliation. Yet, our ultimate goal is to instantiate a high-quality set of correspondences, even if not all necessary input is collected. This is because we can expect that in real-world settings, an expert has a limited effort-budget, and applications require fast setup time, so that waiting for full reconciliation is not a feasible option. To achieve this goal, we develop a pay-as-you-go approach to reconciliation that allows for retrieving a single trusted set of correspondences that satisfies the integrity constraints and maximizes the benefits of expert input at any time.

Our contributions and the outline of this paper can be summarized as follows.

- Section II: We provide a model for probabilistic matching networks. On top of this model, we develop a pay-as-you-go approach to reconciliation that involves an expert asserting the correctness of correspondences. The approach comprises three steps: establishing the probabilistic matching network, reducing the network uncertainty, and instantiating a matching.
- Section III: We show how to establish a probabilistic matching network in the presence of integrity constraints and expert's input. As computing the probabilities that are

assigned to correspondences is computationally costly, we also develop methods to approximate these values.

- Section IV: We provide means to measure and reduce the uncertainty in a probabilistic matching network. Towards this end, we quantify the uncertainty of such a network, define the process of reducing network uncertainty and present the problem of uncertainty reduction under limited effort budget. As a heuristic solution to this problem, we develop a method to order correspondences for which feedback shall be sought using a decision theoretic model.
- Section V: We develop a method that instantiates a matching, a single trusted set of correspondences based on partial user input, making a partial result of data integration available at any time. We formulate the instantiation of such a matching as an optimization problem and propose a heuristic to construct an approximate solution.

The remaining sections are structured as follows. Section VI demonstrates experimental results. Section VII summarizes related work, before Section VIII concludes the paper.

II. Model and Approach

This section starts with a motivating example of a network of schemas. Although the involved schemas are simple, certain questions arise, as we will see, when we attempt to interconnect their attributes. Next, we proceed with the definition of a probabilistic matching network. Finally, we describe a pay-as-you-go approach to reconciliation on top of this network.

A. Motivating Example

Let us consider an online service scenario, where three video content providers EoverI, BBC, and DVDizzy have their own websites to publicize their offerings. Consumers can find the products they want by searching information on the sites (e.g. title, release date). Now the three providers would like to incorporate their websites to broaden the marketplace. Similar product information is stored in their different databases, whose simplified schemas are illustrated in Figure 1. A matching network is created by establishing pairwise matchings between the three schemas in order to facilitate integration scenarios (e.g. support search queries) between the three databases. The figure shows five correspondences c_1, c_2, c_3, c_4, and c_5 which were generated by an automatic matching tool for pairs of schemas. As the involved attribute names are rather similar (date, screenDate, releaseDate, and productionDate), schema matchers often fail to output the correct attribute matches.

Moreover, as the schema matchers only take two schemas as input [7], they ignore natural expectations that one has w.r.t. the entire network of schemas. We can formulate these consistency conditions as integrity constraints. Examples of such constraints have been presented in literature [10], [34], including:

- *One-to-one constraint:* Each attribute of one schema is matched to at most one attribute of any other schema.
- *Cycle constraint:* If multiple schemas are matched in a cycle, the matched attributes should form a closed cycle.

These constraints are natural requirements if one would like to exchange data that is stored in the aforementioned databases. Such network-level constraints describe important consistency conditions; one would like to avoid constraint violations.

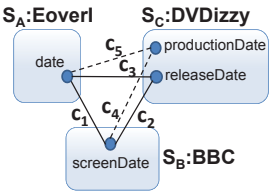

Fig. 1: A matching network of real-world schemas.

Figure 1 shows some violations of these constraints in the set of automatically generated correspondences. The set of correspondences $\{c_3, c_5\}$ violates the *one-to-one constraint*, whereas the set $\{c_2, c_1, c_5\}$ violates the *cycle constraint*.

Problematic correspondences are typically eliminated by reconciliation based on human input: an expert asserts for a given correspondence whether it shall be disregarded or accepted [5]. In the presence of network-level constraints, the order in which an expert gives such input highly influences the overall quality of the network: eliminating certain correspondences early can significantly simplify the reconciliation process. Further, in most scenarios, one cannot expect to get complete feedback on all correspondences identified by a matcher. This calls for a pay-as-you-go approach that allows for retrieving a trusted set of correspondences satisfying the integrity constraints and maximizing the benefits of expert input at any time.

B. Probabilistic Matching Network

This section defines our notion of a probabilistic matching network. An overview of the major concepts is given in Table I. We model a *schema* as a finite set of attributes $s = \{a_1, ..., a_n\}$. Let $\mathcal{S} = \{s_1, ..., s_n\}$ be a set of schemas of a data integration task. Each schema is built of unique attributes (by using unique identifiers), i.e. $s_i \cap s_j = \emptyset$ for all $1 \leq i, j \leq n$ and $i \neq j$. Further, $A_{\mathcal{S}} = \bigcup_i s_i$ is the set of attributes in $\mathcal{S}$. The *interaction graph* $G_{\mathcal{S}}$ represents which schemas need to be matched in the network, i.e. the vertices in $V(G_{\mathcal{S}})$ are labeled by the schemas from $\mathcal{S}$ and there is an edge between two vertices, if the corresponding schemas need to be matched.

An *attribute correspondence* between a pair of schemas $s_1, s_2 \in \mathcal{S}$ is an attribute pair (a, b), such that $a \in s_1$ and $b \in s_2$. The set of *candidate correspondences* $C_{i,j}$ for a pair of schemas $s_i, s_j \in \mathcal{S}$ is a set of attribute correspondences which is typically the outcome of schema matchers [19]. The set of candidate correspondences C for an interaction graph $G_{\mathcal{S}}$ consists of all candidates correspondences for pairs of schemas corresponding to its edges, i.e. $C = \bigcup_{(s_i,s_j)\in E(G_{\mathcal{S}})} C_{i,j}$. Although more complex models for correspondences have been proposed, cf., [18], we focus correspondences modeled as attribute pairs since this model is followed by the vast majority of schema matchers [36], [7].

Integrity constraints for selecting correspondences relate to the expectations that the user has on a valid matching. Let $\Gamma = \{\gamma_1, \ldots, \gamma_n\}$ be a finite set of constraints, which are used to represent the expected consistency conditions. We write $C' \models \Gamma$ to denote that a set of correspondences $C' \subseteq C$ satisfies the constraints in Γ. We do not impose assumptions on the definition of such constraints. For illustration, we rely on the examples of the one-to-one constraint and cycle constraint as defined in [34] and described as part of the motivating scenario in the previous section.

Based on the above notions, we define a network of schemas to be a quadruple $N = \langle \mathcal{S}, G_{\mathcal{S}}, \Gamma, C \rangle$, where $\mathcal{S}$ is a set of schemas (of unique attributes), $G_{\mathcal{S}}$ is an interaction graph, Γ is a set of constraints, and C is a set of candidate correspondences.

The set of candidate correspondences C of N may violate various constraints in Γ and, thus, does not provide a satisfactory result for a data integration task. Instead, we are interested in the *selective matching* M, the set of all correspondences that are correct and satisfy all integrity constraints. To approximate the selective matching, we involve an expert user to assert candidate correspondences. This process is known as reconciliation [34], [5]. The *expert/user input* sought as part of the reconciliation is modeled by a tuple $F = \langle F^+, F^- \rangle$ of assertions, where F^+ and F^- are respectively a set of approved correspondences and a set of disapproved correspondences. That is, upon the user assertion of a correspondence c, we update F by either $\langle F^+ \cup \{c\}, F^- \rangle$ (c is approved) or $\langle F^+, F^- \cup \{c\} \rangle$ (c is disapproved). We exploit user input by concluding that F^+ must be included in M and F^- must be excluded from M. For the remaining candidate correspondences ($C \setminus F^+ \setminus F^-$), a heuristic is used to decide whether they shall be included in a matching.

In this work, we rely on a probabilistic model to develop such heuristics. During the reconciliation, we maintain a set of *probabilities* P that assigns a probability p_c to each candidate correspondence $c \in C$ that indicates how likely correspondence c is to occur in the selective matching. Our probabilistic model acts as a black-box, meaning that it contains all the information given by matchers and user assertions. As such, the user input F is integrated directly in the set of correspondence probabilities P: user assertions are assumed to be always right, so the probabilities of asserted correspondences are either one or zero. Combining the introduced notions, we define a *probabilistic matching network* as a tuple $\langle N, P \rangle$ that represents a single state assumed during reconciliation.

C. Framework for Pay-as-you-go Reconciliation

Constructing the selective matching (the set of correct and consistent correspondences) for a matching network from a set of candidate correspondences can be costly: if a large number of candidate correspondences violate various integrity constraints the reconciliation effort can be considerable. However, many use cases can benefit not only from a fully reconciled match network, but already from a subset of trusted and consistent correspondences, i.e., a matching that approximates the selective matching. In our work, therefore, we study a pay-as-you-go approach to reconciliation.

Figure 2 presents an overview of our framework. We start with a set of candidate correspondences that are generated automatically by schema matchers. Based on these candidate correspondences, we construct a probabilistic matching network

TABLE I: Summary of notations

Symbol	Description
$\mathcal{S}$	A set of schemas
G_S	An interaction graph of a schema matching network
C	An initial set of correspondences generated by matchers
Γ	A set of integrity constraints
$F = \langle F^+, F^- \rangle$	User input of approved and disapproved correspondences
P	A set of probabilities of correspondences
M	The selective matching

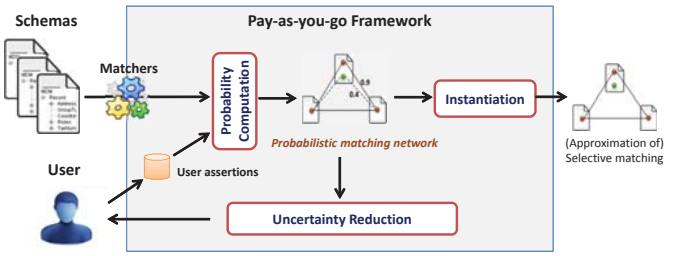

Fig. 2: Framework for pay-as-you-go reconciliation

by means of *Probability Computation*. From a probabilistic matching network, in turn, *Instantiation* automatically derives an approximation of the selective matching that can be used for data integration tasks. The quality of the instantiated matching depends on the degree of uncertainty in the probabilistic matching network. This uncertainty stems from the decision of whether to include correspondences in the instantiated matching. To improve the result of instantiation, *Uncertainty Reduction* selects and ranks candidate correspondences that shall be asserted by a human expert. Based on the obtained user input, the probabilistic matching network is changed by recomputing the probabilities. Consequently, the probabilistic matching network is updated incrementally by the user, whereas an approximation of the selective matching can be generated at any time, thereby achieving pay-as-you-go reconciliation.

Following this general structure, pay-as-you-go reconciliation requires the realization of the following three steps.

Probability Computation. Given a matching network, this step establishes the probability of each correspondence based on information on the integrity constraints of the network and the user input obtained so far. Then, the resulting probabilistic matching network is the underlying model for the other steps since any user input is implicitly incorporated in the probabilities that are assigned to candidate correspondences. The probability computation is described in Section III.

Uncertainty Reduction. To achieve pay-as-you-go reconciliation, the probabilistic matching network is incrementally improved by seeking user input. To guide this uncertainty reduction step, we select and rank candidate correspondences that are shown to user. The ranking criterion is based on information gain that is measured as the amount of uncertainty reduction induced by the user input for a certain correspondence. This step is described in Section IV.

Instantiation. Given a probabilistic matching network, approximation of the selective matching requires making decisions about the selection of correspondences. We realize this step by eliminating a minimal subset of correspondences with low probability from the set of candidate correspondences, such that the remaining correspondences satisfy all integrity constraints. Instantiation is described in Section V.

III. Probability Computation for a Matching Network

Given a network of schemas, this section outlines how a probabilistic matching network is constructed by computing probabilities for the correspondences of the network. We first provide a formal characterization of the probability of a single correspondence (Section III-A). Since the computation of exact

probability values is computationally costly, we then develop techniques to approximate these values (Section III-B).

A. Probability of a Correspondence

Usually, schema matchers assign a confidence value to each candidate correspondence [36]. A confidence value may be interpreted as a probability for the occurrence of the correspondence in the selective matching. Yet, confidence values are not normalized, often unreliable, and unrelated to the application goals [7]. Thus, we take a different approach to compute probabilities for candidate correspondences.

In the context of this work, we assume that the constraints are of paramount importance to the data integration applications. From this starting point, we adopt a model in which a correspondence is more likely to occur in the selective matching, if it is present in many matchings that qualify as approximations of selective matching, i.e., that satisfy the integrity constraints. This property should hold in the presence of user input (approvals or disapprovals of correspondences) that we consider correct. That is, for the computation of probabilities, we consider possible matchings that include all approved correspondences and exclude all disapproved correspondences. We capture the intuition of a matching that qualifies as an approximation of the selective matching with the notion of a matching instance, defined as follows:

Definition 1. ***Matching Instance.*** *Let $\langle S, G_S, \Gamma, C\rangle$ be a network of schemas and $\langle F^+, F^-\rangle$ be user input. A set of correspondences $I \subseteq C$ is a* matching instance*, if*

- *Consistent: I satisfies all integrity constraints (i.e. $I \models \Gamma$) and respects user input (i.e. $F^+ \subseteq I$ and $F^- \cap I = \emptyset$).*
- *Maximal: there does not exists a correspondence $c \in C \setminus (F^- \cup I)$ such that $I \cup \{c\} \not\models \Gamma$.*

Using a Venn diagram, Figure 3 illustrates the relationship of matching instances with candidate correspondences and user input. Any matching instance includes all approved correspondences and excludes all disapproved correspondences. The number of all possible matching instances is at most $2^{|C|}$ as they are subsets of candidate correspondences. F^+ and F^- are disjoint since a correspondence cannot be approved and disapproved at the same time.

Using the notion of a matching instance, we define the probability p_c of a given correspondence c to be proportional to the number of matching instances in which c participates:

$$p_c = \frac{|\{I \in \Omega(F^+, F^-) \mid c \in I\}|}{|\Omega(F^+, F^-)|} \tag{1}$$

where $\Omega(F^+, F^-) = \{I_1, \ldots, I_n\}$, $I_i \subseteq C$, $1 \leq i \leq n$, is the set of all possible matching instances under user input $\langle F^+, F^-\rangle$. Thus, the probability of asserted correspondences is either one or zero, since every matching instance, by definition, includes all approved correspondences and excludes all disapproved correspondences.

Although Equation 1 gives a precise definition of the probability of a correspondence, computing the exact value is costly, especially when user input is incrementally updated. In fact, exact probability computation requires intractably

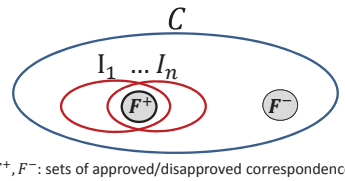

Fig. 3: Relationship between the set of candidate correspondences C, the user input $\langle F^+, F^-\rangle$, and the matching instances $I_1, ..., I_n$

generating all possible matching instances, the number of which is exponential in the total number of correspondences, and verifying them with user input. For example, in the smallest real dataset of our experiments, there are 142 correspondences, resulting in 2^{142} of possible instances in total.

B. Estimating the Probabilities

Since computation of the exact probabilities of correspondences is intractable, we propose a sampling-based approximation to estimate the probabilities. To this end, we generate Ω^* a set of a tractable number of sample matching instances and then consider the probability of a correspondence as the finite limit over Ω^*:

$$p_c = \lim_{\Omega^* \to \Omega(F^+, F^-)} \frac{|\{I \in \Omega^* \mid c \in I\}|}{|\Omega^*|} \tag{2}$$

In order to design an efficient sampling technique for a stream of user assertions, two factors have to be considered:

1) Sample space: it is critical to draw good samples that well capture the exact distribution. Because of the integrity constraints, some correspondences always go together, whereas some others never do. These correlations between correspondences create a complex joint distribution incorporating all possible matching instances.
2) Run time: we consider reconciliation as a pay-as-you-go process where only a few changes are made at a time. Hence, it is not reasonable to re-sample the matching instances from scratch for each user assertion. Instead, we need a technique to maintain a set of preceding samples and update it upon the arrival of a new user assertion.

Addressing these aspects, we rely on a sampling technique that supports non-uniform sampling (approximating the sample space) and view maintenance [8] (improving run time).

Non-Uniform Sampling. Because of the complex joint distribution of matching instances, uniform sampling methods like Monte Carlo are insufficient [23] for probability estimation. Our non-uniform sampling overcomes this limitation by making use of a random-walk strategy and simulated annealing. The role of random-walk is to explore the sample space by generating a next instance from the previous one. Technically, the next instance is computed by randomly adding a correspondence to the current instance and resolving all constraint violations created by this correspondence. However, a random-walk may get trapped in the sample regions with high density [14]. Hence, the role of simulated annealing is to "jump" out of such regions. Due to the dependencies (i.e., integrity constraints) between correspondences in our set-up, the space of matching instances is divided into regions of different magnitude which are not reachable from each other. As a result, combining random-walk and simulated annealing ensures that an instance in a

high-magnitude region should be sampled with a high chance and that an instance in a low-magnitude region should be sampled with a low chance. A formalization of our approach to non-uniform sampling can be found in the Appendix.

View Maintenance. To realize view maintenance, we always keep the set of preceding samples Ω' and update it based on the new assertion of a correspondence c. Then, the set of samples Ω^* is derived as follows, depending on whether c is approved ($\Omega^*(F^+ \cup \{c\}, F^-)$) or disapproved ($\Omega^*(F^+, F^- \cup \{c\})$):

$$\Omega^*(F^+ \cup \{c\}, F^-) = \Omega'(F^+, F^-) \setminus \{I \in \Omega'(F^+, F^-) | c \notin I\}$$
$$\Omega^*(F^+, F^- \cup \{c\}) = \Omega'(F^+, F^-) \setminus \{I \in \Omega'(F^+, F^-) | c \in I\}$$

Maintenance may reduce the number of samples as inconsistent samples are removed, leading to poor estimation of the probabilities. To cope with this limitation, we define a tolerance threshold n_{min}, such that more samples are generated if $|\Omega^*| < n_{min}$. Moreover, if the size of Ω^* is still smaller than n_{min} after two consecutive samplings, it implies that the actual number of all matching instances is smaller than n_{min} and it holds $\Omega^* = \Omega$. Hence, it is not necessary to re-sample since all matching instances have already been generated.

We summarize that our probabilistic model provides a unified way to encode all relevant information on top of the network of schemas. For example, having all probability values equal to one means that the associated correspondences satisfy all integrity constraints and respect user input. As such, the model is well suited as the basis for uncertainty reduction and instantiation of an approximation of the selective matching.

IV. Uncertainty Reduction for a Probabilistic Matching Network

The initial probabilistic matching network computed purely based on the result of automatic matchers shows uncertainty in the sense that the candidate correspondences typically violate a lot of constraints. We reduce network uncertainty with input from an expert that asserts the correspondences. As the amount of expert work is usually limited, we need a technique that can effectively guide the uncertainty reduction. In our setting, the "benefit" is measured as the amount of uncertainty reduction if the correctness of a certain correspondence is asserted.

Following this general idea, below, we first propose a measure for network uncertainty (Section IV-A) and define the process of reducing uncertainty (Section IV-B). This process gives raise to the uncertainty minimization problem for which we provide a formal definition in Section IV-C. A heuristic solution to this problem is presented in Section IV-D.

A. A Measure for Network Uncertainty

In a probabilistic matching network $\langle N, P\rangle$ with $N = \langle \mathcal{S}, G_{\mathcal{S}}, \Gamma, C\rangle$ being a network of schemas, each candidate correspondence $c \in C$ is assigned a probability $p_c \in P$ that represents how likely the correspondences is to be part of the selective matching. This decision, in turn, can be modeled as a binary random variable. Hence, the overall uncertainty of the network is computed as the Shannon entropy [41] over a set of random variables, each one representing the choice whether a certain candidate correspondence occurs in the selective matching. Formally, the Shannon entropy for a probabilistic matching network is defined as follows:

$$H(C, P) = -\sum_{c \in C} [p_c \log p_c + (1 - p_c) \log (1 - p_c)] \quad (3)$$

A network uncertainty $H(C, P) = 0$ means that all probabilities are equal to one or zero; or in other words, there is only one matching instance remaining. In that case, all remaining candidate correspondences, except the disapproved ones, construct a matching that satisfies all the integrity constraints, the selective matching. Hence, our goal in the reconciliation of a probabilistic matching network is to reduce the network uncertainty to zero. Note that 'certain' correspondences – for which we have a probability of zero or one regardless of the origin of this value (user feedback, computation based on matcher result, etc.) – do not contribute to the network uncertainty; i.e. $H(C, P) = H(\{c \in C \mid 0 < p_c < 1\}, P)$.

B. The Process of Reducing Network Uncertainty

Reducing uncertainty in a pay-as-you-go fashion means that the probabilistic matching network is continuously updated by: (1) selecting an attribute correspondence $c \in C$, (2) eliciting user assertion (approval or disapproval) on the correspondence c, and (3) updating the set of probabilities P. That is, by seeking user input for a correspondence, the state of the probabilistic matching network $\langle N, P\rangle$ is changed, leading to the network $\langle N, P'\rangle$, where P' is recomputed from user input as outlined in the previous section. We denote this step of reducing the uncertainty with feedback on correspondence $c \in C$ by $\langle N, P\rangle \xrightarrow{c} \langle N, P'\rangle$. Since the probability computation does not depend on the order of user assertions (see Eq. 1), this notation can be directly lifted to any series of feedback steps on a correspondence set $C' \subseteq C$, i.e., $\langle N, P\rangle \xrightarrow{C'} \langle N, P'\rangle$.

The process of reducing uncertainty may come to a halt once the reconciliation goal is reached. Such a reconciliation goal may be given, for instance, in terms of an effort budget (i.e., the number of user assertions is limited) or a pre-defined threshold for the desired network uncertainty.

A generic procedure of uncertainty reduction is illustrated in Algorithm 1. It takes a probabilistic matching network $\langle N, P\rangle$ and a reconciliation goal δ as input and returns a reconciled probabilistic matching network $\langle N, P'\rangle$. The set of probabilities P' of the reconciled network is initialized based on the current state of the network (line 1). We then proceed as follows (line 2 to 5): First, a correspondence is selected from the set of

Algorithm 1: Generic procedure for reducing uncertainty

input : a probabilistic matching network $\langle N, P\rangle$ with $N = \langle \mathcal{S}, G_{\mathcal{S}}, \Gamma, C\rangle$, a reconciliation goal δ
output: a reconciled probabilistic matching network $\langle N, P'\rangle$

```
// Initialization
1 P' ← P;
2 while not δ do
      // (1) Select a correspondence
3     c* ← select(c ∈ C | 0 < p_c < 1);
      // (2) Elicit and update user input
      // (3) Integrate the feedback
4     Recompute correspondence probabilities P';
5     Recompute network uncertainty H(C, P');
6 return ⟨N, P'⟩;
```

uncertain correspondences (whose probabilities are neither one nor zero). Second, we elicit the assertion for this correspondence and update the user input accordingly. Third, we integrate this feedback by recomputing the set of correspondence probabilities P' and the network uncertainty $H(C, P')$.

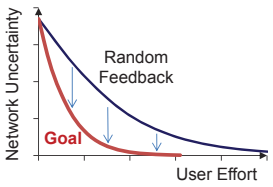

Fig. 4: Uncertainty minimization

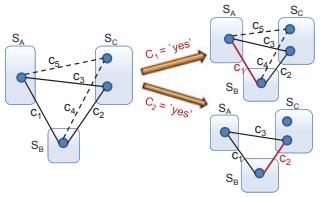

Fig. 5: Effects of corr. ordering

Clearly, there is a trade-off between user effort and network uncertainty: the greater the user input, the less overall uncertainty in the network. Yet, instantiations of Algorithm 1 lead to a different realization of this trade-off. That is, the *select* routine that chooses the correspondences for which feedback shall be sought has to be implemented and affects the degree of uncertainty reduction that is achieved by a certain amount of user input. As a baseline, we consider an expert working without any support tools. This scenario corresponds to the higher curve (random feedback) in Figure 4, in which the *select* routine selects candidate correspondences for assertion in a random order. A more effective implementation of the *select* routine would lower this curve, leading to a higher reduction in network uncertainty for the same amount of user effort compared to the baseline.

Taking up the example introduced earlier, we illustrate below the influence of the selection strategy on network uncertainty.

Example 1. *Figure 5 illustrates the effect of selecting correspondences $C = \{c_1, c_2, c_3, c_4, c_5\}$ for user assertion. According to the aforementioned one-to-one and cycle constraints, there are two matching instances $I_1 = \{c_1, c_2, c_3\}$ and $I_2 = \{c_1, c_4, c_5\}$. The network uncertainty is $H(C, P) = 4$. If c_1 is selected first (either approved or disapproved), I_1 and I_2 are still matching instances since both of them contain c_1; the network uncertainty is unchanged. In contrast, if c_2 is selected first and approved, only I_1 remains as a matching instance since I_2 does not contain c_2; the network uncertainty is 0.*

C. The Uncertainty Minimization Problem

To approach an effective implementation of the routine for selecting correspondences for assertion, we address a concrete reconciliation goal. Since reasonable thresholds for the overall network uncertainty are hard to estimate and user feedback is commonly the bottleneck for reconciliation, we focus on limited budget of user effort. In that case, we would like to minimize network uncertainty under a fixed number of feedback steps. Formally, our objective is defined as follows.

Problem 1. ***Uncertainty Minimization with Limited Effort Budget.*** *Let $\langle N, P\rangle$ be a probabilistic matching network with $N = \langle \mathcal{S}, G_S, \Gamma, C\rangle$, and k be a budget of user effort. The* uncertainty minimization problem with limited effort budget *is the identification of correspondences $C' \subseteq C$ with $|C'| \leq k$, such that $\langle N, P\rangle \xrightarrow{C'} \langle N, P'\rangle$ and $H(C, P')$ is minimal.*

Finding a good selection strategy to solve Problem 1 is challenging. Solving the problem optimally requires investigating all the permutations of all subsets (with size $\leq k$) of candidate correspondences; this is computationally intractable.

D. Uncertainty Minimization by Heuristic Ordering

To address Problem 1 in a non-optimal way, we focus on a heuristic strategy to order the candidate correspondences according to their expected benefit for reducing the network uncertainty. To quantify the potential benefit of knowing whether a correspondence is correct, we follow a decision theoretic approach, cf. [38]. More precisely, we measure the *information gain* of a correspondence c as the expected amount of uncertainty reduction obtained when an approval or disapproval has been given for c. This reduction is computed as the difference between network uncertainty before and after the user asserts c. Since the result of assertion (i.e., approval or disapproval of c) is not known before-hand, the uncertainty of the network obtained after the feedback step is conditioned on c. Let $\langle N, P\rangle$ be a probabilistic matching network with $N = \langle \mathcal{S}, G_S, \Gamma, C\rangle$. Then, we formally define network uncertainty conditioned by the assertions for a particular correspondence as:

$$H(C \mid c, P) = p_c \cdot H(C, P^+) + (1 - p_c) \cdot H(C, P^-) \quad (4)$$

where P^+ are the probabilities of the reconciled network after approval of c, and P^- are the probabilities of the reconciled network after disapproval of c. The information gain of a correspondence c is then computed as the difference of the network uncertainty resulting from approval or disapproval of c, defined as follows:

$$IG(c) = H(C, P) - H(C \mid c, P) \quad (5)$$

Using this measure of information gain, we provide an implementation of the *select* routine of Algorithm 1, in which we choose the correspondences with the highest information gain, i.e., $c^* = \operatorname{argmax}_{c \in \hat{C}} IG(c)$. If the highest information gain is observed for multiple correspondences, one is randomly chosen. It is noteworthy that for all the correspondences whose probabilities are either one or zero (including the ones which the user has already asserted), the information gain is zero. Hence, only uncertain correspondences are qualified for selection, highlighting the encapsulation power of our probabilistic model.

V. Instantiation of an Approximation of the Selective Matching

A distinguished feature of our approach to pay-as-you-go reconciliation is the fact that a matching that approximates the selective matching can be instantiated at all times, even if not all the user input that would be needed for full reconciliation has been collected. Such instantiation is particularly important for applications that value a fast setup time above waiting for full validation [17] and require a deterministic matching that enables querying and aggregating across multiple schemas. In this section, we first formulate the instantiation of an approximation of the selective matching as an optimization problem (Section V-A). Again, this problem turns out to be computationally costly, so that we propose a heuristic to construct a near optimal solution (Section V-B).

A. The Instantiation Problem

Given the set of candidate correspondences of a probabilistic matching network, instantiation refers to the selection of one of the possible matching instances of the network. Ideally, this matching instance is the selective matching. In most cases, however, this matching instance will only approximate the selective matching due to the uncertainty of the network.

To formulate the instantiation problem, we measure the quality of a matching instance with respect to the current state of the probabilistic matching network along two dimensions: its *repair distance* and its *likelihood*. Informally, the former measures the difference between the correspondences of a matching instance and the set of candidate correspondences; the latter indicates the correctness of a set of correspondences given the probabilities of the network. We capture these quality dimensions by two functions defined for a probabilistic matching network $\langle N, P\rangle$ (with N=$\langle S, G_S, \Gamma, C\rangle$) as follows:

- The *repair distance* is a function $\Delta : 2^C \times 2^C \rightarrow \mathbb{N}$ capturing the size of the symmetric difference between two correspondence sets, i.e., $\Delta(A, B) \mapsto |A \setminus B| + |B \setminus A|$ where $A, B \subseteq C$. In our context, we are interested in the repair distance between a matching instance I and the original set of correspondences C. Then, $\Delta(I, C)$ measures the amount of information loss of deriving I from C by eliminating some correspondences to guarantee that the integrity constraints of the network are satisfied.
- The *likelihood* is a function $u : 2^C \rightarrow [0, 1]$ capturing the product of probabilities of a set of correspondences, i.e., $u(A) = \prod_{c \in A} p_c$ where $A \subseteq C$. In our context, considering the likelihood should guide the instantiation towards the selection of correspondences with high probabilities.

Using these measures, we model instantiation of a probabilistic matching network as an optimization problem: we are interested in a matching instance with minimal repair distance w.r.t. the candidate correspondences and maximal likelihood. In the context of schema matching, we consider the repair distance to be more important than the likelihood, since information about correspondences should be preserved as much as possible. Formally, our problem is described as follows.

Problem 2. ***Instantiation.*** *Let $\langle N, P\rangle$ be a probabilistic matching network with $N = \langle \mathcal{S}, G_{\mathcal{S}}, \Gamma, C\rangle$. The* instantiation *problem is the identification of a matching instance I that satisfies the two following conditions, in the descending order of priority:*

i) Minimal repair distance: there is no matching instance I' such that $\Delta(I', C) < \Delta(I, C)$.
ii) Maximal likelihood: there is no matching instance I' with minimal repair distance such that $u(I') > u(I)$.

Note that the instantiation problem is defined only on the probabilities P assigned to correspondences since user assertions are already incorporated implicitly in P. Solving the instantiation problem requires knowledge about the integrity constraints in the network. Unfortunately, even under the simplistic one-to-one constraint and even without the maximal likelihood condition, the instantiation problem is computationally hard.

Theorem 1. *Let $\langle N, P\rangle$ be a probabilistic matching network with $N = \langle \mathcal{S}, G_{\mathcal{S}}, \Gamma, C\rangle$, such that Γ defines the one-to-one constraint. Then, given an integer θ, the problem of deciding whether there exists a matching instance of $\langle N, P\rangle$ with repair distance less than θ is NP-Complete.*

Proof: To prove the NP-completeness of our decision problem, we show that: (i) it is in NP and (ii) it is NP-hard. Given a matching instance I, one can check in polynomial time its repair distance (i.e. $\Delta(I, C)$) is less than θ; so (i) is true. By definition, a matching instance I must not violate the one-to-one constraint; i.e. $\forall c \in I, \nexists c' \in I$ such that c and c' share exactly one common attribute and their remaining attributes belong to the same schema. This case can be represented as an undirected graph $G = (V, E)$ where each vertex $v \in V$ is a correspondence and each edge $e = (v_i, v_j) \in E$ represents a constraint violation between v_i and v_j. G can be constructed in polynomial time by iterating over all the correspondences and creating an edge between any two attributes that match one attribute of a schema to two different attributes of another schema. Finding a matching instance I with minimal repair distance is equivalent to finding a *maximum independent set* (MIS) of G, as no two vertices being adjacent means no violations and $\Delta(I, C) = |C| - |I|$. Since the MIS problem is NP-complete [26], (ii) is true. ■

B. An Instantiation Heuristic

In light of Theorem 1, we consider a heuristic approach to our problem that finds a matching instance I efficiently, at the expense of non-optimality w.r.t. repair distance and likelihood. Developing such a heuristic is far from trivial, given the complex and inter-related dependencies between correspondences that are induced by the integrity constraints.

The approach proposed in this paper is a two-step meta-heuristic algorithm: involving (i) *initialization*, and (ii) *optimization*. The former aims at finding a feasible solution; the latter attempts to optimize this solution. In the initialization step, we greedily pick an initial matching instance among sampled ones. This is motivated by the fact that the space of all possible matching instances is intractably large. Hence, we employ the sampled set of matching instances generated in Section III-B as a starting point for our algorithm. However, since the purpose of sampling is to best capture the exact distribution and because the number of samples is often much smaller than the size of the sample space, we need the optimization step to improve the quality of the initial matching instance. To this end, we apply a randomized local search. The core idea is that we keep exploring the neighbors of recent instances until termination (in our case, an upper bound of iterations), and record the one with the smallest repair distance and the largest likelihood.

The details of our instantiation heuristic are given in Algorithm 2. It takes a probabilistic matching network and a pre-defined upper bound for the number of iterations as input and returns the best matching instance (with smallest repair distance and largest likelihood) it can find during the search. Technically, we begin by greedily picking up a matching instance from sampling (line 1). Starting with the best sample, the local search is repeated until the termination condition is satisfied (line 3). At each iteration, we first generate a set of remaining correspondences and their probabilities. Among

Algorithm 2: An instantiating heuristic

input : a probabilistic matching network $\langle N, P\rangle$ with $N = \langle \mathcal{S}, G_{\mathcal{S}}, \Gamma, C\rangle$, an upper bound for the number of iterations k
output: a matching instance H

```
// Step 1: Initilization - Greedy pickup among samples
1  H ← argmax_{argmax_{I∈Ω(F+,F−)} Δ(I,C)} u(I);
// Step 2: Optimization - Randomized local search
2  I ← H; i ← 0; T ← Queue[k] // a queue with fixed size k
3  while i < k do
       // Fitness proportionate selection
4      ĉ ← RouletteWheel_c({⟨c, p_c⟩ | c ∈ C \ F− \ I});
5      I ← I ∪ {ĉ};
6      T.add(ĉ);
       // Repair matching until constraints are satisfied
7      I ← repair(I, ĉ, F+, Γ) ;
       // Keep track of the optimal instance
8      if Δ(H, C) > Δ(I, C) then
9          H ← I;
10     if Δ(H, C) = Δ(I, C) and u(H) < u(I) then
11         H ← I;
12     i ← i + 1;
13 return H
```

these correspondences, we add one into the current instance I based on Roulette wheel selection [22]. The rational behind this heuristic is that the chosen correspondence has a high chance of being consistent with the others. When a certain correspondence is inserted, it might produce some constraint violations. Thus, the $repair()$ function (defined formally in the appendix) is invoked to eliminate new violations by removing problematic correspondences from I (line 7). However, a correspondence could be added into I and then removed immediately by the repair function, leaving I unchanged, so that the algorithm would be trapped in local optima. For this reason, we employ the Tabu search method [21] that uses a fixed-size "tabu" (forbidden) list of correspondences so that the algorithm does not consider these correspondences repeatedly (line 6). Finally, a matching instance H is returned by evaluating the repair distance and likelihood of matching instances explored so far.

Proposition 1. *Algorithm 2 terminates and is correct.*

Proof: Termination follows from the fact that the termination condition of the routine between line 3 and line 10 can be defined as a constant number k of iterations.
Correctness follows directly from the following points. (1) A new correspondence is not chosen from disapproved correspondences (line 4). (2) When a new correspondence $\hat{c}$ added to I (line 5) causes constraint violations, I is repaired immediately (line 7). (3) H always maintains the instance with smallest repair distance (line 8) and largest likelihood (line 10). Therefore, the algorithm's output is a near optimal matching instance that satisfies all the integrity constraints and respects the user assertions. ■

Finally, we observe that the presented heuristic indeed allows for efficient instantiation. In fact, the algorithm requires quadratic time in the number of candidate correspondences and, thus, is tractable for real datasets.

Proposition 2. *The run time complexity of Algorithm 2 is* $\mathcal{O}(k \times |C|^2)$.

Proof: The most expensive operation in Algorithm 2 is the function $repair()$, which takes at most $\mathcal{O}(|I|^2)$, as outlined in the Appendix. Since $I \subseteq C$ and there are at most k iterations of the local search, we have $\mathcal{O}(k \times |C|^2)$. ■

VI. Experimental Evaluation

This section presents a comprehensive experimental evaluation of the proposed methods using real-world datasets and state-of-the-art matching tools. The results highlight that the presented approach supports pay-as-you-go reconciliation by effective and efficient computation of probabilities. Also, we are able to guide user feedback precisely, observing improvements of up to 48% over the baselines. We demonstrate that the approach improves the quality of instantiated matchings significantly in both precision and recall.

We proceed as follows: We first discuss the experimental setup (Section VI-A). Then, we report on the results of applying the proposed methods for probability computation (Section VI-B), reduction of network uncertainty (Section VI-C), and instantiating of a matching (Section VI-D).

A. Experimental Setup

Datasets and Tools. We relied on four real-world datasets spanning various application domains, from Web forms to business schemas as observed in data marketplaces.
(1) *Business Partner (BP):* The set comprises database schemas that model business partners in enterprise systems.
(2) *PurchaseOrder (PO):* We extracted purchase order e-business schemas from various resources.
(3) *University Application Form (UAF):* We extracted schemas from Web interfaces of American university application forms.
(4) *WebForm:* The schemas for this dataset have been automatically extracted from Web forms using OntoBuilder [37].
These datasets are publicly available [1] and descriptive statistics for the schemas are given in Table II. To generate candidate correspondences, we used two well-known schema matchers (with default parameters), COMA++ [13], [4] and AMC [35]. All experiments ran on an Intel Core i7 system (2.8GHz, 4GB RAM).

TABLE II: Real datasets

Dataset	#Schemas	#Attributes (Min/Max)
BP	3	80/106
PO	10	35/408
UAF	15	65/228
WebForm	89	10/120

TABLE III: Constraint violations

Dataset	# Violations per matcher COMA	AMC
BP	252	244
PO	10078	11320
UAF	40436	41256
WebForm	6032	6367

Integrity Constraints. We consider two well-known constraints, the one-to-one constraint and the cycle constraint, cf., Section II-A. Table III lists the number of constraint violations among the candidate correspondences generated by the matchers. Rather independent of the applied schema matcher, we observe a large number of violations that precludes an exhaustive investigation of the violations by an expert. Hence, there is a clear need for efficient and effective pay-as-you-go reconciliation.

Evaluation Measures. Besides the network uncertainty defined in Equation 3, we rely on the following evaluation measures:

- *Precision & Recall* are measures for the quality of a matching V compared to the selective matching M, i.e., the ground truth, given by the dataset provider: $Prec(V)$=$(|V \cap M|)/|V|$ and $Rec(V)$=$(|V \cap M|)/|M|$.

- *User effort:* To quantify the relative amount of user input, we compute the user effort as the number of asserted correspondences relative to the size of the matcher's output: $E = |F^+ \cup F^-|/|C|$.

B. Evaluations on Probability Computation

For the step of computing the probabilities for a probabilistic matching network, we study the efficiency and effectiveness of the presented sampling technique.

Computation Time. In this experiment, we study the effects of network size (i.e., number of candidate correspondences) on the computation time required for probability estimation. The reporting time is measured by computing the average sampling time over 1000 samples for each setting of network size. Each setting is constructed with a different interaction graph G_S using the Erdős-Rényi random graph model. We then derived the average time over all settings and datasets.

Figure 6 shows the resulting computation time per sample relative to the number of correspondences with values ranging from 2^7 to 2^{12}. Clearly, as the number of correspondences grows, the computation time increases. Yet, absolute numbers are low. For instance, for a network with 4000 candidate correspondences, computation based on 1000 samples takes only $\approx 2 * 10^3 ms = 2s$. Hence, our sampling technique is well applicable for datasets with a large number of correspondences.

Sampling Effectiveness. To measure the sampling effectiveness of our technique, we compute the distance between the sampled probability distribution Q used to approximate the exact probability distribution P. To this end, we use the K-L divergence [27], which is a widely used measure of the information loss of sampling:

$$D_{KL}(P||Q) = \sum_{c \in C} p_c \log \frac{p_c}{q_c} \tag{6}$$

where p_c and q_c are, respectively, the exact and approximate probability of a correspondence c. To allow for better interpretation of the value, we normalize this measure by taking the ratio over the K-L divergence between P and the non-sampled distribution U where $u_c = 0.5$ (following the principle of maximum entropy), which is defined as $KL_{ratio} = D_{KL}(P||Q)/\ D_{KL}(P||U)$.

Figure 7 depicts the results of sampling effectiveness under varying network size, averaged over all datasets. The number of correspondences (x-axis) is limited to 20 since we have to generate all possible matching instances, the number of which is $\mathcal{O}(2^{|C|})$, to compute the exact probability (Equation 1). The KL_{ratio} is given in percentages. We defined the number of samples to be proportional to the number of possible subsets of C, i.e., $2^{|C|/2}$, which is 2^{10} in the case of 20 correspondences.

We observe that although the fraction of samples over all possible matching instances is small (only $\frac{2^{10}}{2^{20}} \approx 0.1\%$ for $|C| = 20$), the KL_{ratio} is less than 2%. In other words, the sampling distribution Q is over 98% better than the baseline (random probability assignment U). This is a strong indicator that Q captures the exact distribution well; and thus, our sampling technique is effective for probability approximation.

Relation between Probability and Correctness. Our approach is based on the hypothesis that a correspondence with high probability is likely to be correct, and vice-versa. To validate this hypothesis, we use the selective matching M from the dataset providers. Figure 8 presents a histogram of the probability distribution in the BP dataset (similar results, omitted for brevity, have been obtained for the other datasets) for correct (existing in M) and incorrect (not existing in M) correspondences. Here, the x-axis depicts the probability ranges and the y-axis measures the frequency in percentages.

We observe that most of the correspondences (more than 75%) have the probability value in the range from 0.5 to 1.0. This is reasonable, since the precision of the generated candidate correspondences in this dataset is about 0.67. Another key finding is that at higher levels of probability, the ratio of correct correspondences over incorrect correspondences is significantly larger. For example, in the $[0.8, 0.9]$ range, there are about 20% correct and about 3% incorrect correspondences; whereas the ratio is about 13%/1% in the $[0.9, 1.0]$ range. This indicates that the probability values indeed reflect the correctness of correspondences.

C. Evaluations on Uncertainty Reduction

We now study the extent to which our approach reduces the network uncertainty, with various budgets of user effort. For each dataset, we generate a complete interaction graph and obtain candidate correspondences using automatic matchers. We simulate the process of reducing network uncertainty where user assertions are generated using the available selective matching.

The influence of the feedback on the quality of the match result is measured in terms of precision and network uncertainty. That is, when increasing the user effort percentage, we quantify the improvements in terms of precision and uncertainty reduction. We compare two ordering strategies to select correspondence for user assertion: (1) *Random*: the random selection of correspondence acts as a baseline for our experiment. (2) *Heuristic:* we select correspondences based on our method exploiting the information-gain (Section IV).

Figure 9 depicts the average result over 50 experiment runs for the BP dataset. Again, we obtained similar results for the other datasets that are omitted for brevity. This result shows a significant reduction of user effort with the *Heuristic* strategy—up to 48% compared to the baseline. For example, to reach a network uncertainty of about 0.1, the $Heuristic$ strategy takes only about 30% user effort, whereas the baseline takes about 75% user effort. Thus, we achieve savings of about 45% of user effort. With only 50% user effort, the $Heuristic$ strategy already achieves a network uncertainty of almost zero, whereas this value is just about 0.4 in the baseline. Regarding precision, the trends of these plots are inversely similar. For example, at 25% user effort with the $Heuristic$ strategy, network uncertainty is about 0.3 and precision is about 0.84. Note that when network uncertainty is zero (i.e. all the integrity constraints are satisfied), the precision is not necessarily guaranteed to be 1.0.

D. Evaluations on Instantiation

Finally, we study the effectiveness of our method for instantiation, i.e., the derivation of a matching from the probabilistic matching network.

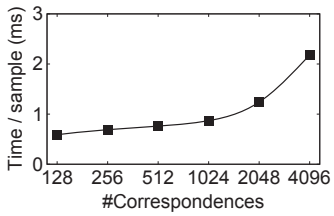

Fig. 6: Effects of network size on computation time of probability estimation

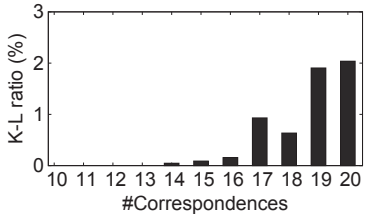

Fig. 7: Sampling efficiency of probability estimation in terms of K-L divergence

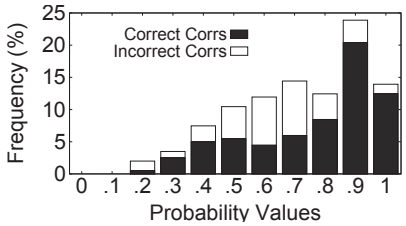

Fig. 8: Relation between probability and correctness of correspondences

Effects of Ordering Strategies. Clearly, the two above ordering strategies used for reducing the network uncertainty have a great influence on the quality of the instantiated matching. We investigate this aspect with an experiment in which, given a predefined user effort (e.g., 5% of all candidate correspondences), we reduce network uncertainty with these strategies (i.e., the *Random* and the *Heuristic*). Then, we compare the results in terms of precision and recall of the matching derived by instantiation according to Algorithm 2.

Figure 10 illustrates the influence of the ordering strategies on quality of the instantiated matching for the BP dataset (again, the other datasets showed the same trend and are omitted for brevity). Here, we varied the budget of user effort (x-axis) from 0% to 15%. A key finding is that our heuristic ordering outperforms the baseline with an average difference of 0.12 (for precision) and 0.08 (for recall). At the beginning (0% user effort), there is no difference between two ordering strategies because no correspondence is selected for user validation. We conclude that our heuristic ordering plays an important role in improving the quality of the matching that approximates the selective matching and is derived by instantiation.

Effects of Maximal Likelihood. Instantiation is guided by the repair distance (number of candidate correspondences that are removed to satisfy the integrity constraints) and the likelihood of a particular matching (cf., Section V-A). We argued that the repair distance shall be minimal in any case to keep us much information on correspondences as possible. Yet, in this experiment, we study the importance of also considering the likelihood for instantiation. To this end, we compare the result of instantiation with and without the likelihood criterion. We quantify the results in terms of precision and recall for the derived matching.

Figure 11 shows the obtained results: the percentage of user effort relative to the quality of the matching measured by precision and recall. We observe that considering the likelihood criterion, indeed leads to a matching of better quality. This result underlines the benefits of our probabilistic model in quantifying the uncertainty of correspondences as well as of the network as a whole.

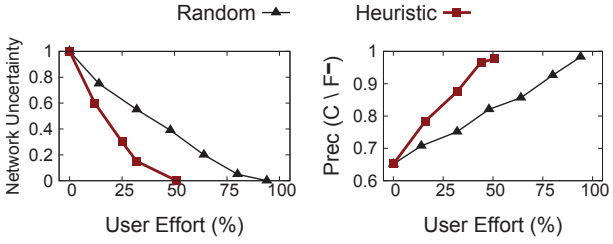

Fig. 9: Effects of heuristic ordering on the uncertainty reduction.

VII. Related Work

We now review the work in schema matching, matching networks, and user feedback that is close to our research.

Schema Matching. Database schema matching is an active research field. The developments of this area have been summarized in several surveys, e.g., [36], [7], [6]. Existing works on schema matching focused mainly on improving the quality parameters of matchers, such as precision or recall of the matches generated. Recently, however, certain researchers have started to realize that the extent to which precision and recall can be improved may be limited for general-purpose matching algorithms. Instead of designing new algorithms, there has been a shift towards matching combination and tuning methods. These works include YAM [16], systematic matching ensemble selection [20] or automatic tuning of the matcher parameters [28]. In this paper, we use results from schema matching tools as input for our approach. Regarding schema matching with uncertainty, the work in [15], [39] studies at data level and pair-wise mapping; whereas, our work relies on a network of schemas and integrity constraints.

Matching Networks. The idea of exploiting a large set of schemas to improve the matching has been studied before. Holistic matching [43] attempted to exploit statistical co-occurrences of attributes in different schemas and use them to derive complex correspondences. Corpus-based matching [29] attempted to use a 'corpus' of schemas to augment the evidence that improves existing matching and exploit constraints between attributes by applying statistical techniques. Network-level constraints were originally considered in [3], [10], in which the establishment of semantic interoperability in large-scale P2P networks was studied. In this paper, we study such integrity constraints in matching networks and use constraint violations as evidence of matching uncertainty.

User Feedback. The post-matching reconciliation process has also received considerable attention in the literature using different approaches. The system in [25], [40], [46] relies on one user only, whereas the frameworks in [47], [31], [33], [32] rely on multiple users. Although our scope involves only a single expert user, our framework is extensible as the underlying probabilistic model is independent of the number of users. Besides, some works [34], [39] also study pay-as-you-go approaches that establish the initial matching and then incrementally improve its quality. Despite the similarities, there are also a number of differences. While the work of [39] focuses on the context of the mediated schema approach, our work studies reconciliation for a network of schemas. While the work of [34] uses a reasoning approach, our work employs a probabilistic approach for reconciliation. While the work

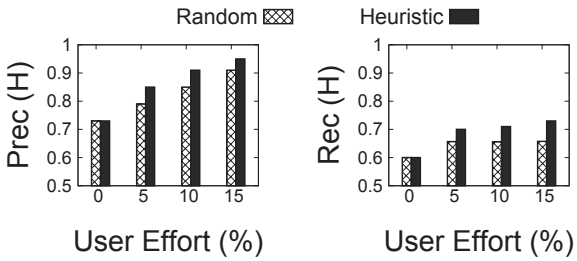

Fig. 10: Effects of correspondence ordering strategies on instantiation. H is the instantiated matching of our algorithm.

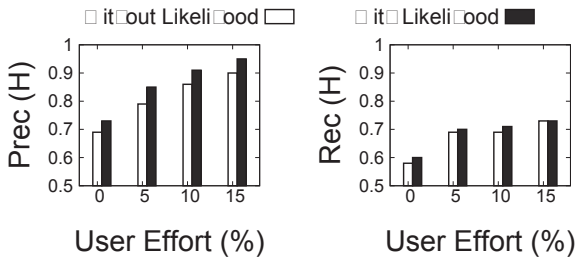

Fig. 11: Effects of the likelihood function on instantiation. H is the instantiated matching of our algorithm.

in [40], [46] gets user feedback implicitly through keywords, our work lets user give input explicitly on the matches.

VIII. Conclusions and Future Work

This paper proposes novel approach that enables pay-as-you-go reconciliation in schema matching networks. We define the notion of a probabilistic matching network as the backbone of the approach. Most importantly, our probabilistic model allows the capture of the uncertainty in the matching network in a systematic way, independent of the used schema matching tools and the data integration tasks that shall be solved. Our approach involves three elementary steps that realize a pay-as-you-go setup: establishing a probabilistic matching network by computing probabilities for correspondences, reducing the network uncertainty with user input, and instantiating a trusted set of consistent correspondences. As such, the approach can be used for supporting data integration at any point in time, while still continuously improving the quality of the instantiated matching by reconciliation of the network. Finally, we presented a comprehensive experimental evaluation of each of the three steps, indicating that the approach is applicable for large, real-world datasets and allows for effective and efficient reconciliation.

Our techniques open up several future directions of research. First, our probabilistic model can be extended to further develop the quality measurement of matching networks. Second, more applications which could be transformed into a matching network shall be devised. This paper focuses on schema matching, yet, the proposed pay-as-you-go approach can be applied to other data integration tasks such as entity resolution, matching of conceptual models, and service discovery.

Acknowledgment. This research has received funding from the NisB project (FP7 - grant number 256955) and the PlanetData project (FP7 - grant number 257641).

References

[1] http://lsirwww.epfl.ch/schema_matching.

[2] http://www.factual.com.

[3] K. Aberer, P. Cudré-Mauroux, and M. Hauswirth, "Start making sense: The Chatty Web approach for global semantic agreements," *JWS*, pp. 89–114, 2003.

[4] D. Aumueller, H.-H. Do, S. Massmann, and E. Rahm, "Schema and ontology matching with coma++," in *SIGMOD*, 2005, pp. 906–908.

[5] K. Belhajjame, N. W. Paton, A. A. A. Fernandes, C. Hedeler, and S. M. Embury, "User feedback as a first class citizen in information integration systems," in *CIDR*, 2011, pp. 175–183.

[6] Z. Bellahsene, A. Bonifati, and E. Rahm, *Schema Matching and Mapping*. Springer, 2011.

[7] P. Bernstein, J. Madhavan, and E. Rahm, "Generic Schema Matching, Ten Years Later," in *VLDB*, 2011.

[8] J. A. Blakeley, P.-A. Larson, and F. W. Tompa, "Efficiently updating materialized views," in *SIGMOD*, 1986, pp. 61–71.

[9] K. Bollacker, C. Evans, P. Paritosh, T. Sturge, and J. Taylor, "Freebase: a collaboratively created graph database for structuring human knowledge," in *SIGMOD*, 2008, pp. 1247–1250.

[10] P. Cudré-Mauroux, K. Aberer, and A. Feher, "Probabilistic Message Passing in Peer Data Management Systems," in *ICDE*, 2006, pp. 41–52.

[11] A. Das Sarma, L. Fang, N. Gupta, A. Halevy, H. Lee, F. Wu, R. Xin, and C. Yu, "Finding related tables," in *SIGMOD*, 2012, pp. 817–828.

[12] G. Di Lorenzo, H. Hacid, H.-y. Paik, and B. Benatallah, "Data integration in mashups," *SIGMOD Rec.*, pp. 59–66, 2009.

[13] H. H. Do and E. Rahm, "COMA - A System for Flexible Combination of Schema Matching Approaches," in *VLDB*, 2002, pp. 610–621.

[14] P. Domingos, D. Lowd, S. Kok, H. Poon, M. Richardson, and P. Singla, "Just add weights: Markov logic for the semantic web," in *URSW*, 2008, pp. 1–25.

[15] X. Dong, A. Y. Halevy, and C. Yu, "Data integration with uncertainty," in *PVLDB*, 2007, pp. 687–698.

[16] F. Duchateau, R. Coletta, Z. Bellahsene, and R. J. Miller, "(not) yet another matcher," in *CIKM*, 2009, pp. 1537–1540.

[17] M. Franklin, A. Halevy, and D. Maier, "From databases to dataspaces: a new abstraction for information management," *SIGMOD Rec.*, pp. 27–33, 2005.

[18] A. Gal, T. Sagi, M. Weidlich, E. Levy, V. Shafran, Z. Miklós, and N. Hung, "Making sense of top-k matchings: A unified match graph for schema matching," in *IIWeb*, 2012.

[19] A. Gal, *Uncertain Schema Matching*. Morgan & Claypool, 2011.

[20] A. Gal and T. Sagi, "Tuning the ensemble selection process of schema matchers," *JIS*, pp. 845–859, 2010.

[21] F. Glover and C. McMillan, "The general employee scheduling problem. an integration of ms and ai," *COR*, pp. 563–573, 1986.

[22] D. E. Goldberg, *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison-Wesley Longman, 1989.

[23] C. Gomes, J. Hoffmann, and A. Sabharwal, "From sampling to model counting," in *IJCAI*, 2007, pp. 2293–2299.

[24] H. Gonzalez, A. Y. Halevy, C. S. Jensen, A. Langen, J. Madhavan, R. Shapley, W. Shen, and J. Goldberg-Kidon, "Google fusion tables: web-centered data management and collaboration," in *SIGMOD*, 2010, pp. 1061–1066.

[25] S. R. Jeffery, M. J. Franklin, and A. Y. Halevy, "Pay-as-you-go user feedback for dataspace systems," in *SIGMOD*, 2008, pp. 847–860.

[26] R. M. Karp, "Reducibility Among Combinatorial Problems," in *Complexity of Computer Computations*, 1972, pp. 85–103.

[27] S. Kullback and R. A. Leibler, "On information and sufficiency," *AOMS*, pp. pp. 79–86, 1951.

[28] Y. Lee, M. Sayyadian, A. Doan, and A. S. Rosenthal, "eTuner: tuning schema matching software using synthetic scenarios," *JVLDB*, pp. 97–122, 2007.

[29] J. Madhavan, P. A. Bernstein, A. Doan, and A. Halevy, "Corpus-based schema matching," in *ICDE*, 2005, pp. 57–68.

[30] D. McAllester, B. Selman, and H. Kautz, "Evidence for invariants in local search," in *AAAI*, 1997, pp. 321–326.

[31] R. McCann, W. Shen, and A. Doan, "Matching Schemas in Online Communities: A Web 2.0 Approach," in *ICDE*, 2008, pp. 110–119.

[32] Q. V. H. Nguyen, X. Luong, Z. Mikls, T. Quan, and K. Aberer, "Collaborative schema matching reconciliation," in *CoopIS*, 2013, pp. 222–240.

[33] Q. V. H. Nguyen, T. T. Nguyen, Z. Miklós, and K. Aberer, "On leveraging crowdsourcing techniques for schema matching networks," in *DASFAA*, 2013, pp. 139–154.

[34] Q. V. H. Nguyen, T. K. Wijaya, Z. Miklos, K. Aberer, E. Levy, V. Shafran, A. Gal, and M. Weidlich, "Minimizing Human Effort in Reconciling Match Networks," in *ER*, 2013.

[35] E. Peukert, J. Eberius, and E. Rahm, "AMC - A framework for modelling and comparing matching systems as matching processes," in *ICDE*, 2011, pp. 1304–1307.

[36] E. Rahm and P. A. Bernstein, "A Survey of Approaches to Automatic Schema Matching," *JVLDB*, pp. 334–350, 2001.

[37] H. Roitman and A. Gal, "Ontobuilder: fully automatic extraction and consolidation of ontologies from web sources using sequence semantics," in *ICSNW*, 2006, pp. 573–576.

[38] S. J. Russell, P. Norvig, J. F. Canny, J. M. Malik, and D. D. Edwards, *Artificial intelligence: a modern approach*. Prentice hall Englewood Cliffs, 1995.

[39] A. D. Sarma, X. Dong, and A. Y. Halevy, "Bootstrapping pay-as-you-go data integration systems," in *SIGMOD*, 2008, pp. 861–874.

[40] M. Sayyadian, H. LeKhac, A. Doan, and L. Gravano, "Efficient keyword search across heterogeneous relational databases," in *ICDE*, 2007, pp. 346–355.

[41] C. E. Shannon and W. Weaver, "A mathematical theory of communication," 1948.

[42] K. P. Smith, M. Morse, P. Mork, M. Li, A. Rosenthal, D. Allen, L. Seligman, and C. Wolf, "The role of schema matching in large enterprises," in *CIDR*, 2009.

[43] W. Su, J. Wang, and F. Lochovsky, "Holistic schema matching for web query interfaces," in *EDBT*, 2006, pp. 77–94.

[44] P. J. Van Laarhoven and E. H. Aarts, *Simulated annealing*. Springer, 1987.

[45] W. Wei, J. Erenrich, and B. Selman, "Towards efficient sampling: exploiting random walk strategies," in *AAAI*, 2004, pp. 670–676.

[46] Z. Yan, N. Zheng, Z. G. Ives, P. P. Talukdar, and C. Yu, "Actively soliciting feedback for query answers in keyword search-based data integration," in *PVLDB*, 2013, pp. 205–216.

[47] A. V. Zhdanova and P. Shvaiko, "Community-Driven Ontology Matching," in *ESWC*, 2006, pp. 34–49.

APPENDIX

Non-Uniform Sampling Algorithm. Here we show the details of our non-uniform sampling in Algorithm 3. The algorithm has two parameters (n, k), four inputs (C, Γ, F^+, F^-), and returns a set of sampling instances Ω^* as output. First, it starts with a trivial sample which contains all the approved correspondences (line 1). Next, it generates the next n samples, each of them being computed from the previous one using random-walk [30], [45] (line 4). Then following the simulated annealing meta-heuristic [44], we consider accepting the next sample with a probability non-uniformly proportional to the number of its correspondences different with those of the current sample (line 7). Since this probability is an exponential function (i.e. $1-e^{-\Delta}$), our sampling is non-uniform. The rationale behind this is that due to the integrity constraints, two particular matching instances are more likely to fall into the same sampling region if they have more common attributes, and vice-versa. In order to avoid being trapped in such sampling regions which are not reachacle from each other, we should make a "jump" with a higher probability if the distance between the current and next sampling instances is larger. The distance between two matching instances is the size of their symmetric difference ; i.e. $\Delta(I_i, I_{next}) = |I_i \setminus I_{next}| + |I_{next} \setminus I_i|$.

Algorithm 3: Non-uniformly sample matching instances

input : a set of correspondences C,
a set of integrity constraints Γ,
a set of approved/disapproved correspondences F^+/F^-,
a number of samples n,
a number of random-walks per sample k
output : A set of sampling instances $\Omega^* = \{I_1, \ldots, I_n\}$

1 $I_0 \leftarrow F^+$; $\Omega^* \leftarrow \emptyset$
2 **for** $i = 1 \rightarrow n$ **do**
3 $I_i \leftarrow I_{i-1}$
// Run random-walk for k steps beginning on I_{i-1}
4 **for** $j = 1 \rightarrow k$ **do**
// Randomly select another correspondence
5 $c \leftarrow Rand(C \setminus F^- \setminus I_i)$
// Add c and repair all constraint violations
6 $I_{next} \leftarrow repair(I_i, c, F^+, \Gamma)$
// Acceptance probability w.r.t. the distance Δ
7 With probability $1 - e^{-\Delta(I_i, I_{next})}$ do $I_i \leftarrow I_{next}$
8 $\Omega^* \leftarrow \Omega^* \cup I_i$
9 **return** Ω^*

Repair Algorithm by Greedy Removal. Algorithm 4 shows the details of our repair heuristic, which implements the $repair()$ function in algorithms 2 and 3. This repair algorithm is used, for a particular instance, to resolve all violations caused by the new correspondence added into that instance. This algorithm's key idea is to greedily remove the correspondences involving new violations, one-by-one, until no violation remains (line 2). In it, we remove the correspondence that causes most constraint violations (line 5 and 6). The insight behind this greediness is that removing correspondences with a high number of violations should be able to minimize the repair distance (Section V).

The complexity analysis is given as follows. First, to check whether an instance I satisfies a set of integrity constraints (line 2), as well as implementing the function $I.getConflict()$ (line 4), we detect all the constraint violations of I. However, as this detection operation is only computed for the added correspondence c (line 1), it takes at most $\mathcal{O}(|I|)$; i.e. all remaining correspondences might be affected. Moreover, the loop between line 2 and line 6 takes place at most $|I|$ times (the worst case in which all correspondences are deleted). As a result, the overall complexity is $\mathcal{O}(|I|^2)$, even though in practice it is likely to be lower since an integrity constraint is often defined upon a small fraction of total correspondences.

Algorithm 4: Repair an inconsistent matching instance

input : an inconsistent (matching) instance I,
an added correspondence c,
a set of approved correspondences F^+,
a set of integrity constraints Γ
output: a consistent matching instance $\hat{I}$

1 $I \leftarrow I \cup \{c\}$
2 **while** I *does not satisfy* Γ **do**
// Get all violations each correspondence c_i involves
3 **for** $c_i \in I \setminus F^+ \setminus \{c\}$ **do**
4 $v_i \leftarrow I.getConflict(c_i, \Gamma)$
// Greedily remove the one with most violations
5 $c^* \leftarrow \text{argmax}_{c_i} |v_i|$
6 $I.remove(c^*)$
7 **return** $\hat{I} \leftarrow I$