Noname manuscript No.
(will be inserted by the editor)

Reconciling Matching Networks of Conceptual Models

Nguyen Quoc Viet Hung [1] · **Matthias Weidlich** [2] · **Nguyen Thanh Tam** [3] · **Zoltán Miklós** [4] · **Karl Aberer** [3] · **Avigdor Gal** [5]

Received: date / Accepted: date

Abstract Conceptual models such as database schemas, ontologies or process models have been established as a means for effective engineering of information systems. Yet, for complex systems, conceptual models are created by a variety of stakeholders, which calls for techniques to manage consistency among the different views on a system. Techniques for model matching generate correspondences between the elements of conceptual models, thereby supporting effective model creation, utilization, and evolution. Although various automatic matching tools have been developed for different types of conceptual models, their results are often incomplete or erroneous. Automatically generated correspondences, therefore, need to be reconciled, i.e., validated by a human expert. We analyze the reconciliation process in a network setting, where a large number of conceptual models need to be matched. Then, the network induced by the generated correspondences shall meet consistency expectations in terms of mutual reinforcing relations between the correspondences. We develop a probabilistic model to identify the most uncertain correspondences in order to guide the expert's validation work. We also show how to construct a set of high-quality correspondences, even if the expert does not validate all generated correspondences. We demonstrate the efficiency of our techniques for real-world datasets in the domains of schema matching and ontology alignment.

Keywords Model matching · Reconciliation · Probalistic Matching

1 Introduction

Conceptual models are an important means to design, analyze, and improve information systems. For instance, database schemas describe the structure of processed entities; ontologies ground their interpretation in terms of well-defined semantics; and process models specify behavioral aspects of information systems. Regardless of the specific model, the creation, utilization, and evolution of conceptual models is supported by manifold concepts and techniques that offer, for instance, re-use driven modeling support, harmonization of model variants, model-based system validation, and effective management of model repositories. Many of these techniques have in common that they rely on *model matching*—the identification of correspondences between the model elements. As such, the accuracy and, therefore, usefulness of techniques supporting the creation, utilization, and evolution of models is highly dependent on the correctness and completeness of model matching.

Model Matching. There is a large body of work on model matching techniques; numerous commercial and academic matching tools, called *matchers*, have been developed to generate correspondences between pairs of database schemas [13, 72,46,45], ontologies [67,26], and process models [22,15, 7]. Even though matchers achieve impressive performance for 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, therefore, model management in general and model matching in particular include a post-matching phase, in which correspondences are reviewed and validated by an expert.

Matching Networks. In this work, we focus on a setting in which matching is conducted for a set of related models. The vast majority of existing matchers generate correspondences between *pairs* of conceptual models. Hence, matching a set of

[1] Griffith University
[2] Humboldt-Universität zu Berlin
[3] École Polytechnique Fédérale de Lausanne
[4] Université de Rennes 1
[5] Technion – Israel Institute of Technology

related models requires the repeated application of a matcher to generate a *matching network*, spanned by the correspondences between elements of the respective models. Such a network, however, shall meet consistency expectations in terms of global integrity constraints. Correspondences generated by a matcher for a pair of conceptual models in isolation may turn out to be problematic when considering the network-level. Correspondences between one pair of models may be inconsistent with correspondences between other models. This raises the question of *how to model matching networks and their integrity constraints*.

The presence of network-level integrity constraints imposes challenges for the manual validation of the matching result obtained for a set of models. The dependencies between correspondences that are induced by constraints are hard to overlook especially in large-scale networks. Also, reconciliation can hardly be shared among several experts since validation of a single correspondence may have far-reaching consequences due to the integrity constraints. Despite these challenges, integrity constraints also open up an opportunity to guide the expert's work by defining the order in which the expert's input is sought. Validation of a particular correspondence may enable reasoning about the correctness of other correspondences, which raises the question of *how to select correspondences to achieve effort-effective reconciliation*.

In real-world settings, an expert has a limited effort-budget and complete reconciliation of a network of correspondences is typically not a feasible option. Rather, there is a need to generate a single trusted set of correspondences that satisfy the integrity constraints and maximize the benefits of the expert input obtained so far. This is the question of *how to instantiate an effective set of consistent correspondences*.

Contributions. In this paper, we address the above research questions based on the notion of a *probabilistic matching network* that assigns probabilities to correspondences in order to represent the uncertainty induced by integrity constraints. Using this model, we develop a pay-as-you-go approach to reconciliation that guides an expert in the validation of correspondences. More specifically, our contributions and the outline of this paper are summarized as follows.

- Section 2: We first discuss different types of conceptual models and their matching problem. We argue that these problems can be generalized into the notion of a matching network and elaborate on the need for reconciliation.
- Section 3: We present a generic formal model for matching networks that includes a probabilistic formulation of network-level integrity constraints.
- Section 4: We provide an overview of our approach for reconciliation that incorporates three elementary steps: the computation of the uncertainty of a matching network, the reduction of the network uncertainty based on expert input, and the instantiation of a matching.
- Section 5: We show how to compute the uncertainty of a matching network using the probabilistic model of a factor graph. To cope with computational challenges, we also introduce methods to deal with large-scale data and incremental updates.
- Section 6: We define the process of reducing the uncertainty of a matching network under a limited budget of expert input as an optimization problem. 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 7: We develop a method that instantiates an effective set of consistent correspondences, referred to as selective matching. We show that this instantiation can be formulated as an optimization problem and propose a heuristic to approximate a solution to this problem.

This paper extends our earlier work on reconciliation of networks of database schemas [64,65]. In particular, we generalize the existing approaches from the domain of databases to conceptual modeling in two dimensions: (1) instead of considering models simply as sets of model elements, we take into account the intra-model structure; (2) we include a notion of soft constraints, for which satisfaction is preferred, but not required. To support the generalized model, this paper presents a novel method to measure the uncertainty of a network based on the notion of a factor graph and a revised instantiation mechanism to obtain a selective matching. We also introduce batch selection of top-k correspondences for expert input, which improves reconciliation efficiency compared to the existing techniques.

The remaining part of the paper is structured as follows. Section 8 demonstrates experimental results. Section 9 summarizes related work, before Section 10 concludes the paper.

2 Background

We first elaborate on matching of conceptual models, before turning to the intuition of matching networks.

2.1 Matching of Conceptual Models

In this work, we explore a generalization of different matching problems between specific types of conceptual models. Below, we outline these matching problems for database schemas, ontologies, and process models.

Schema Matching. Schema matching is the process of generating correspondences between the attributes of two database schemas, for the purpose of some data integration task. 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 [42]. Extraction of information over all regions requires means to query and aggregate across multiple tables, thereby raising the need of an integrated view of the data. Further applications that require schema matching include:

- *Corporate data:* Databases of large enterprises are often developed independently to cater for particular requirements imposed by legal regulations, business models, or value chains. Hence, data resides in multiple sources in an enterprise. To support cross-database queries, the schemas of the databases need to be matched [53,80].
- *P2P networks:* A P2P network is a decentralized and distributed architecture in which participating nodes act as peers that share data. Searching in these networks requires means to query across multiple peers, and schema matching helps to overcome the heterogeneity typically observed in P2P networks [5,6].
- *Cloud platforms:* Cloud applications enable storage and processing of data distributed across PCs, mobile devices, and online services [2,4,3]. To realize a unified view on such data, see [8], schema matching supports the horizontal integration of data across different cloud solutions.

Solutions to the matching problem for database schemas have been developed for decades, see [13] for a recent survey.

Ontology Alignment. Ontologies enable the definition of the semantics and concepts of the Web contents. An ontology can be viewed as a vocabulary to define models of a particular domain that makes explicit the relations between the underlying concepts. Ontologies developed for independent Web sources, however, are heterogeneous due to differences in the syntactic, terminological, and conceptual representation of concepts at these sources. This motivates *ontology alignment*, which aims at generating semantic correspondences between the concepts of ontologies [67] for various use cases:

- *Product catalogs:* In business-to-business applications, online portals and shopping sites store information about their products in electronic catalogs. However, the ontology used to describe products is often designed differently among available sellers. To create a common market place, ontology alignment identifies correspondences between the concepts used to describe products.
- *Web services:* Ontologies provide a rich and precise language to describe the functionality of Web services that expose data via programmable interfaces for information search and discovery [33]. Yet, data and services of different providers are described in terms of diverse ontologies. In this context, ontology alignment helps to identify correspondences between service interfaces for the purpose of web service discovery and comparison [69].

Many matchers developed for schema matching have been adapted to support ontology alignment, e.g., COMA [9], YAM [59], and Harmony [78]. They are complemented by dedicated matchers for ontologies—see the studies conducted by the Ontology Alignment Evaluation Initiative (OAEI) [26].

Process Model Matching. Process models capture the dynamics of a system and play a central role in the design, analysis, implementation, and monitoring of information systems [29]. A process model comprises a set of activities along with causal dependencies for their execution. To compare the behavior described by different process models, process model matching aims at the creation of an alignment between process models by identifying correspondences between their activities [22]. Specifically, the following applications rely on process model matching:

- *System Validation:* Process models are used to document requirements for system implementation, but may also serve as implementation artifacts, defining how enterprise services are orchestrated to support the execution of the process. In this context, the validation of the implementation against the requirements specification can be conducted directly based on the two types of process models [14]. Yet, this requires the identification of corresponding sets of activities in either model, which is supported by process model matching.
- *Variation Management:* In large enterprises, a single process often exists in multiple variations, due to differences among the organizational units in terms of legal requirements, the IT infrastructure, or business strategies. These variants are documented in separate process models and harmonization efforts aim at reducing the number of variation points [87]. Process model matching helps to separate commonalities and differences of process variants.

Matchers for process models adopt similarity measures developed for database schemas and ontologies, yet tailor them to the specifics of process models. Examples include matchers that exploit grammatical structures commonly found in activity labels or the execution semantics of process models. Evaluations of matchers have been published as part of the Process Model Matching Contest [15,7].

2.2 Matching Networks

The above applications, regardless of the type of conceptual model, have in common the need to match sets of related models. Matchers developed for database schemas, ontologies, or process models proceed pair-wise, constructing correspondences between elements (attributes, concepts, or activities, respectively) of two models. Applying these matchers repeatedly for pairs of models induces a matching network that is spanned by correspondences between elements of the respective models.

An Example Matching Network. To give the intuition of matching networks, we consider the scenario of three online video content providers, EoverI, BBC, and DVDizzy. Each provider runs a Web portal, which enables potential customers to search for content (e.g., based on title or release

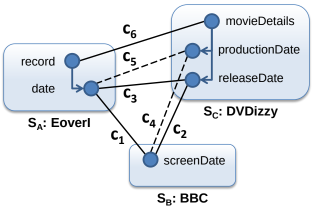

Fig. 1 A matching network of real-world conceptual models.

date). Now, we consider the case that the three providers would like to offer their content via a shared marketplace. Then, as a first step, the databases storing information about the content need to be integrated. The structure of each of these databases is described by a schema and matching these schemas pair-wise creates correspondences between their attributes. Taking the real-world forms of the above mentioned video content providers as a representation of the underlying database schemas, a simplified view on a matching network is shown in Figure 1. It lists some attributes (record, date, screenDate, movieDetails, productionDate, and releaseDate) of the three schemas, some structural dependencies between them (date is part of record, while productionDate and releaseDate are part of movieDetails), and correspondences identified by pair-wise automated matching.

Reconciling Matching Networks. The result of automatic matching of conceptual models is inherently uncertain and it is widely acknowledged that general-purpose matching algorithms cannot be expected to yield results that are always correct. In particular, since matchers identify correspondences between pairs of models, the identification of correspondences is agnostic to global consistency expectations regarding the matching network. These expectations can be formalized as integrity constraints that, unlike those extracted from experts' knowledge [24], are domain-independent. Examples of network-level integrity constraints include the 1-1 constraint and the cycle constraint [20,65].

The 1-1 constraint enforces that each element of one model should be matched to at most one element of any other model. In Figure 1, for instance, the set of correspondences $\{c_3, c_5\}$ violates the 1-1 constraint, since attribute date of schema S_A is matched to both productionDate, and releaseDate of schema S_C. The cycle constraint, in turn, enforces consistency in the sense that any cycle formed by the correspondences is closed [20]. For Figure 1, the set $\{c_1, c_2, c_5\}$ yields a violation: there is a path of correspondences between different elements (attributes productionDate and releaseDate) of a single model (schema S_C).

We further note that the structural relations between the elements of a model impose consistency expectations. For instance, the set of correspondences $\{c_3, c_6\}$ is arguably in line with the relations in model S_A (date being part of record) and model S_C (releaseDate being part of movieDetails): the relations are preserved by the correspondences.

Due to the inherent uncertainty of automatic matching, it was advocated to include a post-matching phase, in which correspondences are reviewed and validated by an expert, a process known as reconciliation [10]. However, most existing approaches study the reconciliation of a results of automatic matching at a pair-wise level; i.e. the correspondences between the elements of a pair of conceptual models are considered in isolation [48,77,88]. These approaches fail to ensure global integrity of the matching network as, e.g., defined by the cycle constraint.

In this paper, we go beyond the common practice of pair-wise reconciliation and include network-level integrity constraints in the reconciliation process to improve the overall matching quality.

3 A Model of Matching Networks

Having introduced the intuition of matching networks in the previous section, this section answers the question of *how to model matching networks and their integrity constraints*.

Conceptual Models. We capture a *conceptual model* as a tuple $\langle A, R \rangle$ with $A = \{a_1, ..., a_n\}$ being a set of *elements* and $R \subseteq A \times A$ as a *structure* relation defined over these elements. As such, we capture the essence of a model in terms of its defining elements and its structure, yet largely abstracting from the peculiarities of diverse types of conceptual models and their representation languages. For instance, elements and the structure relation can be interpreted in terms of database schemas (attributes and functional dependencies), ontologies (concepts and generalization dependencies), and process models (activities and sequential ordering).

A matching network is defined for a finite set of conceptual models $S = \{\langle A_1, R_1 \rangle, \ldots, \langle A_m, R_m \rangle\}$, that are built of unique elements, i.e., $A \cap A' = \emptyset$ for $\langle A, R \rangle, \langle A', R' \rangle \in S$ and $A \neq A'$. Further, $A_S = \bigcup_{\langle A,R \rangle \in S} A$ and $R_S = \bigcup_{\langle A,R \rangle \in S} R$ denote the union of elements and relations in S respectively. In the example in Figure 1, it holds $S = \{S_A, S_B, S_C\}$. Model S_A further comprises elements $A_{S_A} = \{$movieDetails, productionDate, releaseDate$\}$ and its structure relation is defined as $R_{S_A} = \{($movieDetails, productionDate$), ($movieDetails, releaseDate$)\}$.

Interaction Graphs. In many application scenarios, not all given conceptual models are matched pair-wise when constructing a matching network, e.g., due to business requirements or privacy policies. Formally, we capture this aspect by means of an *interaction graph*, an undirected graph $G_S = (V, E)$, such that edges indicate which pairs of models in S need to be matched. For Figure 1, the interaction graph is given as $G_S = (\{S_A, S_B, S_C\}, \{\{S_A, S_B\}, \{S_A, S_C\}, \{S_B, S_C\}\})$.

Correspondences. A *correspondence* represents the (semantic and/or syntactic) equivalence relation between model elements. Formally, we write

$$\mathscr{A} = \bigcup_{\langle A,R\rangle,\langle A',R'\rangle \in S,\ A \neq A'} \ \bigcup_{a \in A,\ a' \in A'} \{a,a'\}$$

to denote the set of all possible correspondences, i.e., all two-sets of elements of distinct models. Then, $\{a,a'\} \in \mathscr{A}$ denotes an individual correspondence. Note that even though many matchers generate solely simple one-to-one correspondences, our formulation does not preclude handling of one-to-many or many-to-many relations, which may be represented by the Cartesian product of the respective elements.

We refer to correspondences generated by automatic matchers as *candidate correspondences* since there is no guarantee that they are indeed correct [35,38]. Given two distinct models $\langle A,R\rangle, \langle A',R'\rangle \in S$, we write $C_{A,A'} \subseteq \mathscr{A}$ to denote the set of all candidate correspondences returned by automatic matchers. In the example in Figure 1, $c_1 = \{\mathsf{date}, \mathsf{screenDate}\}$ is one of the illustrated correspondences.

Integrity Constraints. A finite set $\Gamma = \{\gamma_1, \ldots, \gamma_n\}$ models the integrity constraints that formalize consistency expectations regarding the matching network, such as the 1-1 constraint or cycle constraint. In earlier work, we showed how Answer Set Programming is used as a formal foundation to define these constraints [65]. Such a constraint formalization, however, neglects that the constraints may occasionally be violated due to differences in the abstraction level assumed by the models and correspondences that encode only partial semantic equivalence of model elements. Turning to the above example, for instance, a database schema may encode production details about a movie with a single attribute for the distributor, or a set thereof (one per continent).

Against this background, we formalize network-level integrity constraints based on a probabilistic model. Given a set of candidate correspondences C, this model defines the probability of a constraint being satisfied. However, to capture the dependencies between constraint satisfaction and correspondences on a fine-granular level, this probability is not defined *globally* for the matching network, but *locally* for a particular set of correspondences. That is, an integrity constraint is encoded as a probability $P(\pi \mid C)$, which represents the probability that a set of possible correspondences $\pi \subseteq \mathscr{A}$ satisfies the constraint given a set of candidate correspondences C.

We exemplify this model using several example constraints. We first define generalizations of the two aforementioned constraints, the 1-1 constraint and the cycle constraint. In addition, we present the structure constraint that encodes consistency requirements that are grounded in the structure relation of conceptual models.

- *Generalized 1-1 constraint.* Let $\pi = \{\pi_1, \ldots, \pi_k\} \subseteq \mathscr{A}$ be a set of possible correspondences. Given a set of candidate correspondences C, the generalized 1-1 constraint for π, denoted by $\gamma_{1\text{-}1}(\pi)$, is satisfied with probability

$$P(\gamma_{1\text{-}1}(\pi) \mid C) = \begin{cases} 1 & \text{if } \forall\ \{a,a'\} \in \pi : \\ & |\{c \in C \mid a \in c\}| \leq 1 \\ \Delta & \text{otherwise} \end{cases} \quad (1)$$

  where $\Delta \in [0,1]$ is a relaxation parameter and $\Delta = 0$ yields a hard constraint.
- *Generalized cycle constraint.* Let $\pi = \{\pi_1, \ldots, \pi_k\} \subseteq \mathscr{A}$ be a set of possible correspondences that form a cycle, i.e., $\pi_j = \{a_j, a_{j+1}\}$ for $1 \leq j < k$ and $\pi_k = \{a_k, a_1\}$. Given a set of candidate correspondences C, the generalized cycle constraint for π, denoted by $\gamma_{\circlearrowright}(\pi)$, is satisfied with probability

$$P(\gamma_{\circlearrowright}(\pi) \mid C) = \begin{cases} 1 & \text{if } \pi \subseteq C \\ 0 & \text{if } |\pi \setminus C| = 1 \\ \Delta & \text{otherwise} \end{cases} \quad (2)$$

  where $\Delta \in [0,1]$ is a relaxation parameter modeling the probability of compensating errors (i.e., two or more incorrect correspondences yielding a correct cycle).
- *Structure constraint.* Let $S = \{\langle A_1,R_1\rangle, \ldots, \langle A_m,R_m\rangle\}$ be a set of conceptual models. Furthermore, let $\pi = \{\pi_1, \ldots, \pi_k, \pi'_1, \ldots, \pi'_k\} \subseteq \mathscr{A}$, $k < m$, be a set of possible correspondences, such that:
  - the correspondences form two paths, $\pi_j = \{a_j, a_{j+1}\}$ and $\pi'_j = \{a'_j, a'_{j+1}\}$ for $1 \leq j \leq k$;
  - the paths visit the same models, $a_j, a'_j \in A_j$ for $1 \leq j \leq k$;
  - the structure relation in the first and last model is inconsistent, $(a_1, a'_1) \in R_1$ and $(a'_{k+1}, a_{k+1}) \in R_{k+1}$.

  Given a set of candidate correspondences C, the structure constraint for π, denoted by $\gamma_R(\pi)$, is satisfied with probability

$$P(\gamma_R(\pi) \mid C) = \begin{cases} 0 & \text{if } \pi \subseteq C \\ 1 & \text{if } |\pi \setminus C| = 1 \\ \Delta & \text{otherwise} \end{cases} \quad (3)$$

  where $\Delta \in [0,1]$, again, is a relaxation parameter modeling the probability of compensating errors.

Relaxation parameters Δ of constraints may be provided by application experts or be set based on an adaptive learning strategy. While this parameter configuration is not the focus of this paper, we outline a basic learning strategy in Appendix A.

Matching Networks. Combining the above notions, we define a *matching network* as a tuple $N = \langle S, G_S, \Gamma, C\rangle$, where S is a set of conceptual models, G_S is an interaction graph defined between these models, Γ is a set of integrity constraints, and C is a set of candidate correspondences.

4 Pay-As-You-Go Reconciliation

Given a matching network, formalized as $N = \langle S, G_S, \Gamma, C\rangle$ according to the above model, reconciliation aims at identifying a *selective matching* $M \subseteq C$, i.e., a subset of the candidate correspondences that are most likely correct and consistent in terms of the specified integrity constraints. A selective matching can be seen as an approximation of the unknown ground truth based on the current knowledge about the correctness of correspondences.

Following common models for reconciliation [10], we assume the presence of an expert that reviews and validates correspondences. However, we argue that, due to network-level integrity constraints, the order in which an expert validates correspondences influences the result quality and the required reconciliation effort. In addition, it is not realistic to assume that an expert can validate all candidate correspondences. To answer the questions of *how to select correspondences to achieve effort-effective reconciliation* and *how to instantiate an effective set of consistent correspondences*, we present an approach to pay-as-you-go reconciliation that is grounded in the model of probabilistic matching networks.

Below, we first clarify the notion of expert input obtained in the reconciliation process, before defining the notion of a probabilistic matching network. Finally, we give an overview of our overall reconciliation approach.

Expert Input. We model *expert input* as a tuple $U = \langle U^+, U^-\rangle$ of assertions, where $U^+ \subseteq \mathscr{A}$ and $U^- \subseteq \mathscr{A}$ are sets of approved and disapproved correspondences, respectively. That is, after expert input has been sought for a candidate correspondence $c \in C$, the assertions U are updated, yielding either $\langle U^+ \cup \{c\}, U^-\rangle$ (c is approved) or $\langle U^+, U^- \cup \{c\}\rangle$ (c is disapproved). In this work, we assume a feedback model, where expert input is considered to be correct, such that correspondences in U^+ must be included in a selective matching M, whereas correspondences in U^- must be excluded.

Probabilistic Matching Networks. Our approach relies on the notion of a probabilistic matching network to decide in which order expert input shall be sought and which of the candidate correspondences that have not been validated ($C \setminus U^+ \setminus U^-$) shall be included in a selective matching. A *probabilistic matching network* is a tuple $\langle N, P\rangle$ that extends a matching network $N = \langle S, G_S, \Gamma, C\rangle$ with a probability model P. The latter assigns a probability $P(c)$ to each candidate correspondence $c \in C$, indicating how likely it is that c is correct. This model integrates the user assertions U: since expert input is assumed to be correct, the probabilities of asserted correspondences are one or zero.

Overview of the Approach. Our approach to pay-as-you-go reconciliation is illustrated in Figure 2. We start from a set of candidate correspondences that are generated by automatic matchers. Based on these candidate correspondences, we construct a probabilistic matching network by means of *Probability Computation*. As will be explained in detail in Section 5, each candidate correspondence is assigned a correctness probability based on information on the integrity constraints of the network and the user input obtained so far.

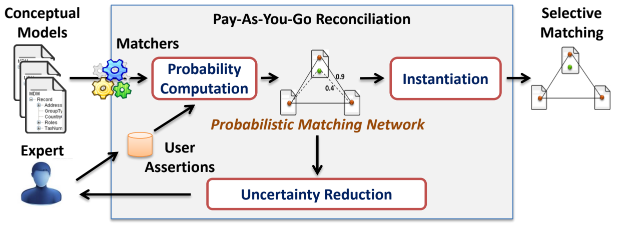

Fig. 2 Overview of our approach to pay-as-you-go reconciliation

The model of a probabilistic matching network encodes for each correspondence the uncertainty of whether the correspondence is part of the matching result. *Uncertainty Reduction*, therefore, aims at increasing the quality of the matching result by selecting and ranking candidate correspondences that shall be asserted by user. As will be described in Section 6, the ranking is based on a decision theoretic model, measuring the information gain of expert input for a certain correspondence.

Instantiation, in turn, constructs a selective matching from the probabilistic matching network. As will be shown in Section 7, this step retains a maximal subset of candidate correspondences that are likely to be correct and consistent with the integrity constraints.

Based on the obtained user input, the probabilistic matching network is updated by recomputing the probabilities of candidate correspondences. Consequently, the network is updated incrementally by the user, whereas a selective matching can be instantiated at any time, thereby achieving pay-as-you-go reconciliation.

5 Probability Computation

Construction of a probabilistic matching network requires the computation of probabilities: for each correspondence, we need to determine the probability of it being correct. Initially, these probabilities are computed solely from the set of candidate correspondences produced by automatic matchers. Often, matchers assign a so called confidence value to each candidate correspondence [72]. However, it has been observed that these confidence values are not normalized, often unreliable, dependent on the used matcher, and are unrelated to application goals [13]. Consequently, a confidence value cannot reliably be interpreted in terms of a probability.

In the context of this work, therefore, we ground the computation of probabilities for correspondences in the integrity constraints defined for the matching network. From this starting point, we adopt a model in which a correspondence is a random variable. Then, integrity constraints express dependencies between these random variables and assertions obtained by expert input are evidence for their truth values. Below, we first show how this idea is formalized using the model of a factor graph, before we turn to the actual computation of probabilities for the correspondences.

5.1 The Factor Graph of a Matching Network

We capture the dependencies between the random variables of correspondences, integrity constraints, and expert input by means of a probabilistic graphical model, namely a *factor graph* [51]. In general, a factor graph establishes a relation between functions (called factors) that are defined over potentially overlapping sets of random variables. The model enables self-configuration when new information becomes available, which is an important asset to support pay-as-you-go reconciliation: With the arrival of new expert inputs, conceptual models, or candidate correspondences, the model is updated incrementally by adding variables and factors.

A factor graph is a bipartite graph $\langle V, F, E\rangle$ where V is a set of random variables or evidence, F is a set of functions (factors), and $E \subseteq \{\{v, f\} \mid v \in V, f \in F\}$ are undirected edges. A set of random variables V and a set of factors F fully characterizes a factor graph. The definition of the edges relates each factor $f(v_1, \ldots, v_d) \in F$ to the random variables over which it is defined, i.e., $\{f, v_i\} \in E$ for $v_i \in V$, $1 \leq i \leq d$.

In our context, there are three types of random variables representing candidate correspondences, constraints, and expert inputs. We overload notation and use C, Γ, and U to refer to the actual correspondences, constraints, and expert inputs, as well as the associated random variables, i.e., $V = C \cup \Gamma \cup U$ defines the variable nodes of the factor graph. Further, the model includes correspondence factors f_C, constraint factors f_Γ, and expert factors f_U to encode relations between the variables, i.e., $F = f_C \cup f_\Gamma \cup f_U$ defines the factor nodes of the factor graph.

Correspondence Variables. As mentioned above, each correspondence $c \in C$ is assigned a random variable, also denoted by $c \in \{0,1\}$, that indicates the correctness of the correspondence (1 denotes correctness).

Constraint Variables (Evidence). We refer to a set of correspondences that violate or satisfy an integrity constraint as constraint evidence. Each of these sets is assigned a variable node $\Pi \subset C$ in the factor graph.

Expert Variables. Expert input $u \in U$ is directly considered as an (observed variable) $u \in \{0,1\}$ (1 denotes approval).

Correspondence Factors. Each correspondence variable c is associated with a prior-distribution factor $f_c : \{c\} \to [0,1]$ that is determined either in a training phase or stems from automatic matchers (e.g., based on the confidence value assigned by these matchers). If no information is available, we start with $f_c(c) = 0.5$ following the maximum entropy principle. The set of correspondence factors is $f_C = \bigcup_{c \in C} f_c$.

Constraint Factors. A constraint factor node cf connects a correspondence to the constraint violations it involves. As a result, the correspondences are dependent on each other through multiple factors, which encodes their stochastic dependency. For illustration, we rely on the three aforementioned constraints. Let $\pi = \{\pi_1, \ldots, \pi_k\} \subseteq \mathscr{A}$ bet a set of possible correspondences. Then, constraint factors are defined as follows:

- The function for a factor that represents the *generalized 1-1 constraint* π is defined as:

$$cf_{\gamma_{1\text{-}1}(\pi)}(c_1, \ldots, c_k) = P(\gamma_{1\text{-}1}(\pi) \mid C) \tag{4}$$

  where $c_i \in \pi \cap C$ and $c_i \neq c_j$ for $1 \leq i, j \leq k$ and $i \neq j$.
- If the correspondences in π form a cycle (see, Equation 2), the factor representing the *generalized cycle constraint* is defined as:

$$cf_{\gamma_{\circlearrowright}(\pi)}(c_1, \ldots, c_k) = P(\gamma_{\circlearrowright}(\pi) \mid C) \tag{5}$$

  where $c_i \in \pi \cap C$ and $c_i \neq c_j$ for $1 \leq i, j \leq k$ and $i \neq j$.
- If the correspondences in π form two inconsistent paths (see, Equation 3), the factor representing the *structure constraint* is defined as:

$$cf_{\gamma_R(\pi)}(c_1, \ldots, c_k) = P(\gamma_R(\pi) \mid C) \tag{6}$$

  where $c_i \in \pi \cap C$ and $c_i \neq c_j$ for $1 \leq i, j \leq k$ and $i \neq j$.

Note that probability computation is also applicable for hard constraints (studied in our previous work [64]), as the probability of these constraints becomes zero or one.

Expert Feedback Factors. To incorporate expert input, each correspondence variable c is connected with an expert input u via a factor node f_u. This factor directly encodes the response of the expert to accept or reject the respective correspondence:

$$f_u(c, u) = \begin{cases} 1 & \text{If } u = 1 \wedge c = 1 \\ 1 & \text{If } u = 0 \wedge c = 0 \\ 0 & \text{Otherwise} \end{cases} \tag{7}$$

Under a different model for expert input, this formulation can be adapted to include experts that, for instance, are not fully reliable, so that their assertion may be wrong. However, such extended formalizations are beyond the scope of this work and we refer interested readers to [60, 63] for possible implementations of such adapted models.

Taking up the example matching network of Figure 1, Figure 3 illustrates a respective factor graph. It comprises variables (shown as circles) for five correspondences $c_1, \ldots, c_5$ and four variables for evidence $\Pi_1, \ldots, \Pi_4$, each representing a set of possible correspondences that violate one of the

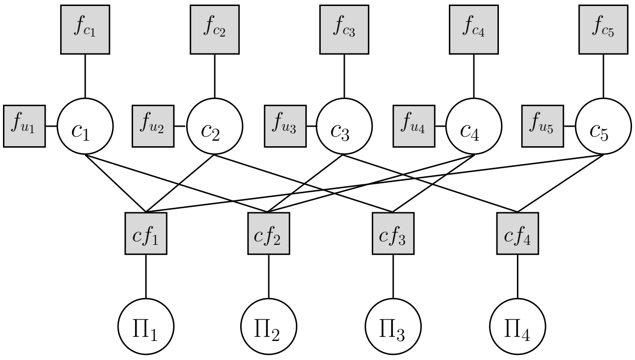

Fig. 3 Factor graph representation of a matching network

integrity constraints. For instance, $\Pi_1 = \{c_1, c_2, c_5\}$. Expert variables are not visualized. The figure also illustrates various factors, i.e., $f_{c_1}()$ is the correspondence factor assigning a prior for the correctness of correspondence c_1 to the respective random variable. Factor $uf_1()$ links the latter to the observed variable of expert input for this correspondence (the variable is not visualized). Further, $cf_1()$ is a constraint factor that connects the set of possible correspondences Π_1 to the variables of the correspondences c_1, c_2, and c_5, implying that $\{c_1, c_2, c_5\}$ is a violation of a constraint—specifically, the cycle constraint is violated in this example. Note that the correspondence c_6 is not visualized here since it is not involved in any constraint violation; i.e., it is represented by an isolated variable with only a correspondence factor $f_{c_6}()$ and a feedback factor $uf_6()$ connected to it.

5.2 Computing the Probabilities for Correspondences

The model of a factor graph enables us to compute the certainty of a correspondence in a matching network. This computation exploits the (marginal) probabilities of the random variables representing the correctness of correspondences. Following the model introduced above, correspondence variables are binary, so that $P(c = 1)$ (or $P(c)$ for short) is the probability that a correspondence $c \in C$ is correct. The computation of this probability is grounded in the correlations defined by the factor functions that relate the random variables to each other.

Various techniques are available to compute probabilities in a factor graph, most commonly *belief propagation* or *sampling*. The former considers the (un)certainty as information that is propagated through the factor graph and relies, for instance, on message-passing algorithms [51]. Yet, it has been observed that belief propagation converges slowly if the graph is large and contains cycles [92]. When reconciling matching networks, the number of variables grows quickly and cyclic dependencies become the rule, rather than the exception. To cope with large and dense factor graphs, we resort to sampling to find the most probable values of random variables. Specifically, Gibbs sampling proved to be a highly efficient and effective mechanism for factor graphs [92].

In brief, given a probabilistic matching network $\langle N, P\rangle$, probability computation yields, for each $c \in C$, a probability value $P(c)$ indicating the likelihood of the respective correspondence being correct. Note, again, that the factor graph model encodes all the information given by automatic matchers (in terms of priors) and expert input (in terms of assertions).

Scalability Considerations. In general, computing probabilities in the whole factor graph is an expensive computational task, which stands in contrast to the low response times needed in reconciliation based on expert input. However, reconciliation is an incremental process, meaning that only a few changes have to be incorporated at a time. Hence, recomputing the whole graph is typically not necessary once new expert input has been received. Following this line, an implementation of probability computation shall incorporate the following two techniques to achieve scalability:

- *Incremental Gibbs sampling:* The computation of Gibbs sampling can be adapted to proceed incrementally [55], rejuvenating the existing probability values in light of new data. Then, the new data is propagated to neighboring nodes, while applying a decay function. The latter limits the consequences of a local change, and thus makes sampling faster.
- *Network modularity:* Even experts are typically overwhelmed when presented with a complete matching network. However, regardless of the type of conceptual model, we observed that, in practice, these models are typically built from disjoint groups of model elements. Thus, candidate correspondences often relate to disjoint subsets of model elements, so that graph decomposition techniques [43] can be applied to decompose a large network into smaller ones to handle them more efficiently. In cases the disjoint groups are still large, the number of correspondences and constraint violations can grow fast, making it intractable to construct the factor graph. To handle such cases, the decomposition strategies presented in our previous work [61] for partitioning a matching network into smaller parts can be applied. The resulting partial networks can be efficiently transformed into factor graphs, with the trade-off of information loss regarding constraint violations that involve correspondences between elements of conceptual models in different partitions. Yet, such information loss may be minimized by tracing the decomposition back to hypergraph partitioning [61]. We note, though, that even for the large datasets used in our experimental evaluation (see Section 8), the construction of a factor graph did not turn out to be problematic and no decomposition of the matching network had to be applied.

6 Uncertainty Reduction in a Matching Network

Initially, a probabilistic matching network is constructed purely based on the result of automatic matchers. This network shows uncertainty in the sense that it comprises many candidate correspondences, or sets thereof, that violate integrity constraints. This uncertainty is reduced by incorporating the input from an expert that asserts the correspondences.

Since expert input is a scarce resource, we need a technique that can effectively guide the uncertainty reduction. To this end, we can exploit the fact that integrity constraints induce dependencies between correspondences. Hence, the selection of candidate correspondences for validation by an expert can be driven by the expected benefit of having asserted the correctness of a certain correspondence.

Against this background, we address the question of *how to select correspondences to achieve effort-effective reconciliation*. We take into account that selection of a single correspondence for validation by an expert (as done in [64]) incurs high overhead in terms of probability computation. Consequently, we show how to select a set of top-k candidate correspondences for eliciting expert input.

The section first proposes a measure for network uncertainty. Next, we define the process of reducing uncertainty in a probabilistic matching network, which gives raise to the uncertainty minimization problem. Finally, we introduce a heuristic solution based on the factor graph representation of the probabilistic matching network.

6.1 A Measure for Network Uncertainty

Given a probabilistic matching network $\langle N,P\rangle$ with $N = \langle S, G_S, \Gamma, C\rangle$, we recall that each candidate correspondence $c \in C$ is assigned a correctness probability $P(c)$. To measure the overall uncertainty of the network, we resort to the Shannon entropy [79] over these probabilities, defined as follows:

$$H(C,P) = -\sum_{c\in C} \left(P(c)\log P(c) + (1-P(c))\log(1-P(c))\right) \tag{8}$$

An overall uncertainty of the network of $H(C,P) = 0$ means that all probabilities assigned to candidate correspondences are equal to one or zero. Therefore, a selective matching can be constructed directly: it comprises all candidate correspondences that are assigned a correctness probability of one. Hence, our goal when reconciling a probabilistic matching network is to reduce the network uncertainty to zero.

We note that 'certain' correspondences—those having a probability of zero or one regardless of the origin of this value (automatic matcher, expert input, probability computation using the factor graph)—do not contribute to the network uncertainty: $H(C,P) = H(\{c \in C \mid 0 < P(c) < 1\}, P)$.

6.2 The Reconciliation Process

Reducing uncertainty in a pay-as-you-go fashion means that the probabilistic matching network $\langle N,P\rangle$, $N = \langle S, G_S, \Gamma, C\rangle$, is continuously updated by:

(1) selecting a set of candidate correspondences $D \subseteq C$;
(2) eliciting user assertion (approval or disapproval) on the correspondences D; and
(3) updating the probability model P.

That is, by seeking user input for correspondences, the state of the probabilistic matching network $\langle N,P\rangle$ is changed, leading to the new probabilistic matching network $\langle N,P'\rangle$, where P' is computed as detailed in Section 5. We denote this step of reducing uncertainty with expert input on a set of correspondences $D \subseteq C$ by $\langle N,P\rangle \xrightarrow{D} \langle N,P'\rangle$. The process of reducing uncertainty may come to a halt once a reconciliation goal is reached. Such a reconciliation goal may be given, for instance, in terms of an effort budget (i.e., the number of assertions by an expert is limited) or a predefined threshold for the desired network uncertainty.

A generic procedure of uncertainty reduction, referred to as the reconciliation process, 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 network $\langle N,P'\rangle$. The algorithm works as follows: First, a set of top-k correspondences is selected from the candidate correspondences. To this end, we postulate a function $select$ that choses a subset D of the candidate correspondences C, taking into account the current probabilistic model P'. Second, expert input is sought for these correspondences, represented here by a function $expert$ that returns $\langle U^+, U^-\rangle$, the sets of approved and disapproved correspondences, respectively. Third, the expert input is integrated by updating the probabilistic model P' and recomputing the network uncertainty $H(C,P')$. Function $update$ implements the approach to probability computation introduced in Section 5.

Clearly, there is a trade-off between expert effort and network uncertainty: the more expert input is sought, the less overall uncertainty is expected in the network. Yet, instantiations of Algorithm 1 lead to a different realization of this trade-off, depending on the implementation of function $select$. As a baseline, we consider an expert working without any supporting tools. This scenario corresponds to the higher curve (random feedback) in Figure 4, in which the $select$ routine selects candidate correspondences in a random order. A more effective implementation of $select$ would lower this curve, leading to a higher reduction in network uncertainty for the same amount of expert input compared to the baseline.

To approach an effective implementation of function $select$, we address a concrete reconciliation goal. Since reasonable thresholds for the overall network uncertainty are hard to estimate and expert input is commonly the bottleneck for reconciliation, we focus on limited budget of expert

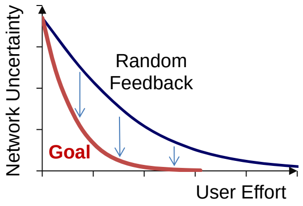

Fig. 4 Uncertainty minimization

Algorithm 1: Reconciliation process

```
input  : a probabilistic matching network $\langle N, P\rangle$ with $N = \langle S, G_S, \Gamma, C\rangle$,
         a reconciliation goal $\delta$
output : a reconciled matching network $\langle N, P'\rangle$
$P' \leftarrow P$;
while not $\delta$ do
    // (1) Select a set of top-$k$ correspondences
    $D \leftarrow select(C, P')$;
    // (2) Elicit expert input
    $\langle U^+, U^-\rangle \leftarrow expert(D)$ ;
    // (3) Integrate the expert input
    $P' \leftarrow update(N, P', \langle U^+, U^-\rangle)$ ;
    Recompute network uncertainty $H(C, P')$;
return $\langle N, P'\rangle$;
```

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 $N = \langle S, G_S, \Gamma, C\rangle$ be a matching network, m be a budget of expert effort, and k be the number of selected candidate correspondences per iteration of the reconciliation process ($k \ll m$). The problem of uncertainty minimization with limited effort budget is the identification of correspondences $C' \subseteq C$ with $|C'| \leq m \cdot k$, 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 the problem of uncertainty minimization is challenging: in order to obtain an optimal solution, all permutations of all subsets (with size $\leq m \times k$) of candidate correspondences needed to be investigated. This is computationally intractable.

6.3 Uncertainty Minimization by Heuristic Ordering

To avoid exploration of an exponential number of candidate correspondences, we use a heuristic strategy to materialize the *select* function in Algorithm 1. It orders candidate correspondences according to their expected benefit to reduce network uncertainty.

6.3.1 Measuring the Expected Benefit

To quantify the potential benefit of a set of correspondences, we follow a decision theoretic approach, cf. [74]. Specifically, we measure the *information gain* of a set of correspondences D as the expected amount of uncertainty reduction obtained when the correspondences are approved or disapproved. This reduction is computed as the difference between network uncertainty before and after the expert validates D. Since the result of validation (i.e., approval or disapproval of each correspondence in D) is not known before-hand, the uncertainty of the network obtained after integrating the expert input is conditioned on D.

Formally, we define network uncertainty conditioned by the assertions for a particular set of correspondences as:

$$H(C \mid D, P) = \sum_{c_1, \ldots, c_k \in D} P(c_1, \ldots, c_k) \cdot H(C, P | \{c_1, \ldots, c_k\}) \tag{9}$$

Here, $\sum_{c_1, \ldots, c_k \in D}$ denotes the sum over all combinations of binary values of the random variables representing the correspondences in D; $P(c_1, \ldots, c_k)$ is the probability of a particular value combination, and $P|\{c_1, \ldots, c_k\}$ denotes the probabilistic model for a particular value combination.

The information gain of a set of correspondences D is then computed as the difference of the network uncertainty resulting from expert input on correspondences in D:

$$IG(D) = H(C, P) - H(C \mid D, P) \tag{10}$$

Using this measure of information gain, we implement function *select* of the reconciliation process (Algorithm 1), such that we choose the top-k correspondences with the maximal information gain. That is, *select* is defined as

$$\underset{D \subseteq C, |D| = k}{\text{argmax}} \; IG(D) \tag{11}$$

If the maximal information gain is observed for multiple subsets, one is randomly chosen. We note that for all correspondences with probabilities being one or zero, the information gain is zero. Hence, only uncertain correspondences are qualified for selection.

However, to solve Equation 11, we need to tackle two major complexity issues, when the number of correspondences is large and k is large. First, calculating Equation 9 requires iterating over all possible combinations of binary values of the random variables of the k given correspondences, i.e. $k!$ binary permutations. Second, a naive solution to find the maximal set of candidate correspondences requires iterating over all possible subsets of correspondences with size k, i.e. exploration of $\binom{n}{k}$ subsets. To overcome these two challenges, we resort to an *approximate computation* of the benefit and a *greedy algorithm* for the actual selection, respectively.

6.3.2 Approximate Benefit Computation

The computation of the (joint) information gain of a set of correspondences is intractable. Therefore, we resort to an alternative utility function. It combines the individual benefit of each correspondence and a redundancy penalty that takes into account dependencies between correspondences.

Individual benefit. The expected benefit of a single correspondence is calculated as follows:

$$IG(c) = H(C,P) - H(C \mid \{c\},P). \tag{12}$$

In this case, according to Equation 9, $H(C \mid \{c\},P)$ collapses to $P(c) \cdot H(C,P|c=1) + (1-P(c)) \cdot H(C,P|c=0)$, which is tractable.

Following this idea, one could select, one-by-one, the correspondences with maximal information gain, i.e., the correspondences for which the probability is close to 0.5. However, this method may be non-optimal due to the complex joint distribution of random variables representing the correspondences. Simply accumulating candidate correspondences with maximal individual information gain does not necessarily mean that the resulting set has the maximal information gain. In terms of the factor graph used to compute the correctness probabilities of correspondences, the truth value of a correspondence variable also influences its surrounding variables and factors.

Redundancy Penalty. Neglecting the dependencies between correspondences variables in the factor graph may yield redundant validation effort. For example, in Figure 3, without prior information, the correspondences c_3 and c_5 are highly uncertain. However, seeking expert input for both of them is redundant: approval/disapproval of $c3$ can directly lead to disapproval/approval of c_5, due to the 1-1 constraint.

Therefore, selecting a candidate correspondence shall aim at low information overlap. Formally, the redundancy of a set of correspondences is quantified as:

$$R(D) = \sum_{c,c' \in D} IG(c)M(c,c')IG(c') \tag{13}$$

where $M(c,c') = \frac{1}{Z}|\{cf \in F|c,c' \in cf\}|$ is a correlation matrix that is grounded in the number of constraint factors that are connected to both c and c' and normalized to the unit interval by $Z = \max_{c,c' \in C}|\{cf \in F|c,c' \in cf\}|$. Intuitively, integrating the redundancy penalty helps to avoid selecting correspondences that are correlated, to each other and to those already validated in previous iterations of the reconciliation process.

The Approximated Benefit. The utility function to approximate the benefit of a set of correspondences combines the two aforementioned measures. Yet, it also weights the importance of correspondences. Here, the idea is that correspondences stemming from a large group of dependent correspondences have a high chance to propagate information. To exploit this effect, we define $q(c) = \sum_{c' \in C} M(c,c')IG(c')$ as the importance of correspondence c.

Putting it all together, we define the utility function as:

$$Q(D) = w \sum_{c \in D} q(c)IG(c) - \sum_{c,c' \in D} IG(c)M(c,c')IG(c') \tag{14}$$

where $w \in R^+$ is a positive weight parameter to balance the terms related to individual benefit and redundancy. The implementation of function *select* of the reconciliation process then becomes:

$$\underset{D \subseteq C, |D|=k}{\text{argmax}} \; Q(D). \tag{15}$$

Finally, we note that our notion of utility to approximate the benefit of a set of correspondences shows two intuitive properties, namely monotonicity and submodularity.

Property 1 (Monotonicity) Q is monotonic: for all $D_1 \subseteq D_2 \subseteq C$, we have: $Q(D_1) \leq Q(D_2)$.

Proof (Sketch) With $w \geq 2$, we have:

$$\begin{aligned}
Q(D_1 \cup D_2) - Q(D_1) &= w \sum_{c \in D_2} q(c)IG(c) - (\sum_{c \in D_2, c' \in D_1} IG(c)M(c,c')IG(c') \\
&+ \sum_{c \in D_1, c' \in D_2} IG(c)M(c,c')IG(c') + \sum_{c,c' \in D_2} IG(c)M(c,c')IG(c')) \\
&= w \sum_{c \in D_2} IG(c) \sum_{c' \in V} M(c,c')IG(c') - (2 \sum_{c \in D_1, c' \in D_2} IG(c)M(c,c')IG(c') \\
&+ \sum_{c,c' \in D_2} IG(c)M(c,c')IG(c')) \geq 2 \sum_{c \in D_2} IG(c) \sum_{c' \in V} M(c,c')IG(c') \\
&- (2 \sum_{c \in D_1, c' \in D_2} IG(c)M(c,c')IG(c') + \sum_{c,c' \in D_2} IG(c)M(c,c')IG(c')) \\
&= 2 \sum_{c \in D_2} (\sum_{c' \in V} M(c,c')IG(c') - \sum_{c' \in D_1 \cup D_2} M(c,c')IG(c')) \\
&= 2 \sum_{c \in D_2} \sum_{c' \notin D_1 \cup D_2} M(c,c')IG(c') \geq 0
\end{aligned}$$

which completes the proof of monotonicity.

Property 2 (Submodularity) Q is submodular: for all $D \subseteq V$ and $c_1, c_2 \in (V \setminus D)$, we have:

$$Q(D \cup \{c_1\}) + Q(D \cup \{c_2\}) \geq Q(D \cup \{c_1, c_2\}) + Q(D)$$

Proof (Sketch) We have:

$$Q(D \cup \{x\}) - Q(D) = wq(x)IG(x) - 2IG(x) \sum_{c \in D} M(x,c)IG(c) + IG^2(x)$$

Then with $w > 0$, we have:

$$\begin{aligned}
&Q(D \cup \{c_1\}) + Q(D \cup \{c_2\}) \geq Q(D \cup \{c_1, c_2\}) + Q(D) \\
&\Leftrightarrow Q(D \cup \{c_1\}) - Q(D) \geq Q(D \cup \{c_2\} \cup \{c_1\}) - Q(D \cup \{c_2\}) \\
&\Leftrightarrow wq(c_1)IG(c_1) - 2IG(c_1) \sum_{c \in D} IG(c)M(c,c_1) + IG^2(c_1) \\
&\geq wq(c_1)IG(c_1) - 2IG(c_1) \sum_{c \in D \cup \{c_2\}} IG(c)M(c,c_1) + IG^2(c_1) \\
&\Leftrightarrow 2IG(c_1)IG(c_2)M(c_1,c_2) \geq 0
\end{aligned}$$

which completes the proof of submodularity.

While computation of the utility function Q is tractable, the induced optimization problem (Equation 15) is not.

Theorem 1 *Solving Equation 15 is NP-complete.*

Proof $Q(D)$ is a submodular set function. Maximization of submodular set functions is known to be NP-complete [58].

6.3.3 Greedy Correspondence Selection

To avoid the complexity of solving the optimization problem based on the utility function, we approximate its solution with a greedy selection strategy. Exploiting the aforementioned properties of the utility function Q, i.e., monotonicity and submodularity, we construct a greedy algorithm with a provably near-optimal result as follows.

For any monotone, submodular function f with $f(\emptyset) = 0$ it is known that an iterative algorithm selecting the element e with maximal value of $f(D \cup \{e\}) - f(D)$ with D as the elements selected so far has a performance guarantee of $(1 - 1/e) \approx 0.63$ [57]. Applying this procedure in our context, we iteratively expand the selection of correspondences by adding the correspondence through k iterations. At each of the k iterations, we traverse all non-validated correspondences to identify the correspondence c^* to maximize $Q(D' \cup \{c^*\})$, where D' is the set of correspondences selected in previous iterations.

The time and space complexity of this heuristic selection strategy are $\mathcal{O}(|C|^2 + k|C|)$ and $\mathcal{O}(|C|^2)$, respectively. The quadratic term $|C|^2$ in time and space complexity stems from the calculation of the correlation matrix $M(.,.)$. The linear term $k|C|$ is explained by k iterations, each requiring consideration of the whole set of candidate correspondences to compute Q and select c^*.

7 Instantiation of the Selective Matching

A distinguishing feature of our approach to pay-as-you-go reconciliation is the fact that a matching can be instantiated at all times, even if the matching network is not fully reconciled. The instantiation of such a matching is particularly important for applications that value a fast setup time above waiting for full validation [34]. Also, various applications explicitly require a deterministic matching, e.g., to query or aggregate a collection of conceptual models, see Section 2.

In this section, we first formulate an optimization problem for the instantiation of a selective matching, i.e., a matching that comprises candidate correspondences that are most likely correct and consistent in terms of the specified integrity constraints. Since this problem turns out to be computationally costly, we then propose a heuristic-based algorithm to construct a near optimal solution.

7.1 The Instantiation Problem

Given a matching network, a set of candidate correspondences is called a *matching instance*. Ideally, a matching (instance) is the ground truth, which would be obtained after all candidate correspondences have been validated by means of expert input. Yet, this is impractical in most cases, so that the uncertainty in a probabilistic matching network induces a set of matching instances that approximate the ground truth. To assess the quality of a particular matching instance $I \subseteq C$ of a probabilistic matching network $\langle N, P \rangle$ with $N = \langle S, G_S, \Gamma, C \rangle$, we consider three dimensions:

- *Violation Degree:* A matching instance of high quality should be likely to satisfy the integrity constraints. Formally, we capture this requirement by the *violation degree* per integrity constraint $\gamma \in \Gamma$, a function $v_\gamma : 2^C \rightarrow [0,1]$ that denotes the probability of a set of correspondences to violate the constraint. We exemplify the definition of this function for the 1-1 constraint formalized in Section 3. Let $I \subseteq C$ be a matching instance. Then, the violation degree is the probability of violating the constraint of the matching instance, i.e., $v_{\gamma_{1\text{-}1}}(I) = (1 - P(\gamma_{1\text{-}1}(I) \mid I))$.
- *Size:* A matching instance should relate a large number of model elements to each other. Given a matching instance $I = C$, its size in terms of the number of contained correspondences $|I|$ is a straightforward measure for this quality dimension.
- *Likelihood:* A matching instance should comprise correspondences that are likely to be correct. Therefore, we consider the *likelihood* of matching instances, which is defined by a function $u : 2^C \rightarrow [0,1]$. Given a matching instance $I = \{c_1, \ldots, c_k\} \subseteq C$, it captures the joint probability of the correspondences, $u(I) = P(c_1, \ldots, c_k)$. It is worth noting that using the factor graph representation of a probabilistic matching network, this joint probability is computed via the associated factors using message passing algorithms or Gibbs sampling.

Using these measures, we model instantiation of a matching as an optimization problem. In model matching, we prioritize the violation degree, since any output presented to a user should be consistent. However, given the probabilistic nature of constraints, we adopt some tolerance threshold $\theta \in [0,1]$ for the violation degree. From all matching instances that show a violation degree that is less than the threshold, we identify one that has maximal size and maximal likelihood. Formally, the problem is described as follows.

Problem 2 (Matching Instantiation) Let $\langle N, P \rangle$ with $N = \langle S, G_S, \Gamma, C \rangle$ be a probabilistic matching network and $\theta \in [0,1]$ be a tolerance threshold. The problem of *matching instantiation* is the identification of a matching instance $I \subseteq C$ that satisfies the following conditions, in the descending order of priority:

i) θ-satisfaction for all constraints: for all constraints $\gamma \in \Gamma$, it holds that $v_\gamma(I) \leq \theta$.
ii) Maximal size: for all matching instances $I' \subseteq C, I' \neq I$, that show θ-satisfaction for all constraints holds that $|I| \geq |I'|$.
iii) Maximal likelihood: for all matching instances $I' \subseteq C$, $I' \neq I$, that show θ-satisfaction for all constraints and maximal size holds that $Q(I) \geq Q(I')$.

An exact solution to the matching instantiation problem is called a *selective matching*. Note that the problem is grounded solely in the correctness probabilities of correspondences since expert input is incorporated in the probabilistic model.

Solving the problem of matching instantiation requires knowledge about the integrity constraints in the network. Unfortunately, even under the simplistic 1-1 constraint and even without the maximal likelihood condition, the instantiation problem is computationally hard.

Theorem 2 *Let $\langle N,P \rangle$ with $N = \langle S,G_S,\Gamma,C \rangle$ be a probabilistic matching network, such that $\Gamma = \{\gamma_{1\text{-}1}\}$ defines only the 1-1 constraint. Then, given a tolerance threshold $\theta \in [0,1]$ and an integer $\delta \in \mathbb{N}$, the problem of deciding whether there exists a matching instance of $\langle N,P \rangle$ that shows θ-satisfaction of Γ and is of size larger than δ 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 whether its size is larger than δ and whether I shows θ-satisfaction of Γ (using the factor graph representation). So (i) is true. Next, we argue that there is a polynomial time reduction of the *maximum independent set* (MIS), which is NP-complete [49], to our problem. MIS requires the identification of a maximal set of vertices in a graph $G = (V,E)$ such that no two vertices are adjacent. We construct a probabilistic matching network as follows: each vertex $v \in V$ is a correspondence. An edge $\{v_i,v_j\} \in E$ is represented by a pair of distinct correspondences that do not show θ-satisfaction of the 1-1 constraint, i.e., $\{v_i,v_j\} \in E$ iff $v_i,v_j \in \{c \in C \mid (1 - P(\gamma_{1\text{-}1}(c) \mid I)) > \theta\}$. This construction requires iterating over all pairs of nodes, i.e., polynomial time. Then, solving our decision problem yields a solution to MIS, so that (ii) is true.

7.2 A Heuristic Solution to Matching Instantiation

In light of Theorem 2, we develop a heuristic solution to find an approximation of a selective matching efficiently. The approximate solution is found in polynomial time, yet may be non-optimal w.r.t. size and likelihood.

Developing a heuristic solution for the problem of matching instantiation is challenging due to the complex dependencies between correspondences that are induced by the integrity constraints. Some correspondences always go together, whereas others are mutually exclusive because of the integrity constraints. These dependencies create a non-uniform joint distribution incorporating all possible matching instances. Our approach, therefore, is to rely on a randomized local search. The main idea is to keep exploring the neighbors of recent matching instances until termination (in our case, an upper bound of iterations), and record the one with the best size and likelihood.

Algorithm 2: Heuristic matching instantiation

```
   input  : a probabilistic matching network $\langle N,P \rangle$ with $N = \langle S,G_S,\Gamma,C \rangle$,
            an upper bound for the number of iterations $k$,
            a tolerance threshold $\theta$
   output : a matching instance $H$
   // Initilization
1  $H \leftarrow \{c \in C \mid P(c) = 1\}$ ;
2  $I \leftarrow H$;
3  $i \leftarrow 0$ ;
4  $T \leftarrow \emptyset$ ;
5  while $i < k$ do
       // Fitness proportionate selection
6      $\hat{c} \leftarrow RouletteWheel_c(\{\langle c,P(c) \rangle \mid c \in (C \setminus I \setminus T)\})$;
7      $I \leftarrow I \cup \{\hat{c}\}$;
8      $T \leftarrow T \cup \{\hat{c}\}$;
       // Adjust matching, so that it shows $\theta$-satisfaction
9      $I \leftarrow adjust(P,I,\hat{c},\Gamma,\theta)$ ;
       // Keep track of the best instance
10     if $|H| < |I|$ then $H \leftarrow I$ ;
11     if $|H| = |I| \wedge Q(H) < Q(I)$ then $H \leftarrow I$ ;
12     $i \leftarrow i + 1$;
13 return $H$
```

Overview. Our approach to heuristic matching instantiation is formalized in Algorithm 2. It takes a probabilistic matching network, an upper bound for the number of iterations, and a tolerance threshold as input. It returns the best matching instance that is found during the search. Technically, the algorithm starts with a trivial matching instance that contains all correspondences for which the assigned probability is equal to one. This instance typically comprises a small number of correspondences. The instance is then extended until the termination condition is satisfied (line 5). In each iteration, we first consider a set of remaining correspondences and their probabilities. One of these correspondences is added to the current matching instance I based on Roulette wheel selection [40]. Once a correspondence has been added, the current matching instance may no longer show θ-satisfaction for all constraints. Therefore, the *adjust* function (defined below) potentially removes problematic correspondences from I to ensure θ-satisfaction (line 9). However, a correspondence could be added to I and then removed immediately by the *adjust* function. In that case, I would be left unchanged and the algorithm would be trapped in a local optima. Therefore, we employ the Tabu search method [39] that uses a 'tabu' (forbidden) set of correspondences, so that the algorithm does not consider these correspondences repeatedly (line 8). Finally, a matching instance H is returned by evaluating the size and likelihood of matching instances explored so far.

Adjusting a Matching Instance. Algorithm 3 details function *adjust* in Algorithm 2, which adjusts a matching instance that does not show θ-satisfaction. The key idea is to greedily remove correspondences that are involved in likely constraint violations, until the matching instance shows θ-satisfaction (line 2). We do so by first extracting all subsets of correspondences of the matching instance that contain the correspondence that has just been added ($\hat{c}$) and are problematic in terms of the constraints (line 3). We then identify correspondences that may be removed, because their correctness probability is less than one and the have not been just added (line 4). For these correspondences, we count how often they are part of subsets of correspondences that do not show θ-satisfaction (line 5) and remove the correspondences for which the largest count is obtained (line 7). The greediness of this approach is motivated by the idea that correspondences that are likely to cause the absence of θ-satisfaction of the original matching instance are removed first. This way, many of the correspondences of the matching instance are retained.

Algorithm 3: Adjusting a matching instance

```
input  : a probabilistic model $P$,
         a matching instance $I$,
         an added correspondence $\hat{c}$,
         a set of integrity constraints $\Gamma$,
         a tolerance threshold $\theta$
output : a matching instance $\hat{I}$ that shows $\theta$-satisfaction for all constraints $\Gamma$
1 $\hat{I} \leftarrow I$
2 while $\exists\, \gamma \in \Gamma : v_\gamma(\hat{I}) > \theta$ do
      // Get all sets of problematic correspondences
         containing $\hat{c}$
3     $W \leftarrow \{C' \subseteq \hat{I} \mid \hat{c} \in C' \wedge \exists\, \gamma \in \Gamma : v_\gamma(C') > \theta\}$;
4     $I_P \leftarrow \{c \in \cup_{C' \in W} \mid c \neq \hat{c} \wedge P(c) < 1\}$;
      // For each correspondence, count in how many
         problematic sets it occurs
5     for $c \in I_P$ do $b_c \leftarrow |\{C' \in W \mid c \in C'\}|$ ;
      // Greedily remove the one that occurs in the largest
         number of problematic sets
6     $c^* \leftarrow \mathrm{argmax}_{c \in I_P}\, b_c$;
7     $\hat{I} \leftarrow \hat{I} \setminus \{c^*\}$;
8 return $\hat{I}$
```

Properties of the Heuristic Solution. For the presented heuristic solution, we provide guarantees in terms of correctness and runtime performance.

Guarantee 1 *Algorithm 2 terminates and is correct.*

Proof Termination follows directly from the upper bound k for the number of iterations in Algorithm 2 and the fact that in each iteration in Algorithm 3, one correspondence is removed, but none is added.
Correctness follows directly from the following points. (1) A new correspondence is chosen from probable correspondences (line 6). (2) When a correspondence $\hat{c}$ added to I (line 7) leads to absence of θ-satisfaction of the matching instance I, I is adjusted immediately (line 9). (3) H always maintains the instance that is best in terms of size and likelihood. Therefore, the algorithm's output is a near-optimal solution to the problem of matching instantiation.

Finally, we observe that the presented heuristic solution indeed allows for efficient instantiation of a matching for the aforementioned integrity constraints (see Section 3). For the 1-1 constraint and cycle constraint, the algorithm requires quadratic time in the number of candidate correspondences, which, as we demonstrate in our experimental evaluation, is tractable.

Guarantee 2 *For the 1-1 constraint and the cycle constraint, the runtime complexity of Algorithm 2 is $\mathcal{O}(k \times |C|^2)$.*

Proof We start with the function *adjust*, i.e., Algorithm 3. First, all sets of correspondences that contain $\hat{c}$, but do not show θ-satisfaction are extracted. For the considered 1-1 constraint and cycle constraint, we note that the constraints in these sets are necessarily connected. Hence, the sets can be derived by depth-first-search, starting with correspondence $\hat{c}$. Visiting each correspondence at most once, this yields a runtime complexity of $\mathcal{O}(|I|)$. Moreover, there are at most $|I|$ iterations (in the worst case, all correspondences are removed). As a result, the overall complexity is $\mathcal{O}(|I|^2)$.
Finally, the most expensive operation in Algorithm 2 is the function *adjust*, which has a runtime complexity of $\mathcal{O}(|I|^2)$. Since $I \subseteq C$ and there are at most k iterations of the local search, we arrive at $\mathcal{O}(k \times |C|^2)$.

For the structure constraint, the size of the structure relation also influences the runtime complexity. However, in practice, the number of entries in the structure relation is significantly less than the number of correspondences $|R_S| \ll |C|$, which means that the complexity is actually close to the one obtained for the other constraints.

Guarantee 3 *For the structure constraint, the runtime complexity of Algorithm 2 is $\mathcal{O}(k \times |C| \times (|C| + |R_S|))$.*

Proof Similar to the proof of Guarantee 2, we also need to perform depth-first-search. The only difference is that now we have to include the relations in R_S as connections, which yields a runtime complexity of $\mathcal{O}(|I| + |R_S|)$ for the correspondence $\hat{c}$. Since remaining operations are similar, we arrive at the overall complexity $\mathcal{O}(k \times |C| \times (|C| + |R_S|))$.

8 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. We are able to precisely guide expert users, so that the amount of expert input needed for reconciliation is reduced to 50% or less compared to baseline solutions. 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 8.1). Then, we report on the results of applying the proposed methods for probability computation (Section 8.2), reduction of network uncertainty (Section 8.3), and instantiation of a matching (Section 8.4).

8.1 Experimental Setup

Datasets and Matching 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 dataset 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 [73].
(5) *Conference (OAEI):* This is a collection of 16 ontologies about conference organization, from the 'conference' track of Ontology Alignment Evaluation Initiative (OAEI) 2014 [26].
These datasets are publicly available [1] and descriptive statistics for the schemas are given in Table 1. To generate candidate correspondences for schema matching datasets (BP, PO, UAF, WebForm), we used two well-known schema matchers (with default parameters), COMA++ [23,9] and AMC [70]. For ontology matching datasets (OAEI), we use the AML matcher [32] due to its good performance in the 2014 edition of the OAEI [26].

Integrity Constraints. In our experiments, we consider two of the aforementioned constraints, the 1-1 constraint and the cycle constraint, cf., Section 3. Table 2 lists the number of candidate correspondences (or sets thereof) generated by the matcher for which the constraint satisfaction probability is less than one. Rather independently of the applied dataset and matching tool, we observe a large number of problematic correspondences, which precludes an exhaustive investigation by an expert. Hence, there is a clear need for efficient and effective pay-as-you-go reconciliation.

Table 1 Datasets

Dataset	#Models	#Elements (Min/Max)
BP	3	80/106
PO	10	35/408
UAF	15	65/228
WebForm	89	10/120
OAEI	16	23/140

Table 2 Problematic correspondences

Dataset	# Correspondences ($v_\gamma < 1$)		
	COMA	AMC	AML
BP	252	244	N/A
PO	10078	11320	N/A
UAF	40436	41256	N/A
WebForm	6032	6367	N/A
OAEI	N/A	N/A	9352

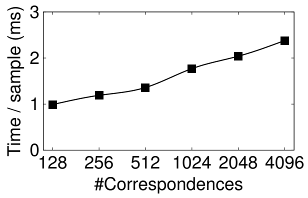

Fig. 5 Effects of network size on probability computation

Evaluation Measures. In addition to the network uncertainty as defined in Equation 8, we rely on the following evaluation measures:

- *Precision & Recall* are measures for the quality of a matching V (i.e. a set of correspondences) compared to the exact matching R (e.g. a set of referenced correspondences given by the dataset provider): $Prec(V)=(|V \cap R|)/|V|$ and $Rec(V)=(|V \cap R|)/|R|$.
- *User effort:* To quantify the relative amount of expert input, we compute the effort as the number of asserted correspondences relative to the size of the matcher's output: $E = |U^+ \cup U^-|/|C|$.

Experimental Environment. All results have been obtained on an Intel Core i7 system (2.8GHz, 4GB RAM). Factor graph modeling and computing have been conducted using Elementary [92].

8.2 Evaluation of the Probability Computation

For the step of computing the correctness probabilities of correspondences in a matching network, we study the efficiency and effectiveness of the presented approach that exploits a factor graph.

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 computation. We use the Elementary framework [92] that computes probabilities in a factor graph by Gibbs sampling. The reported 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 5 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 $\approx 2.4\text{ms} \cdot 10^3 = 2.4\text{s}$. Hence, the presented approach is well applicable for datasets with a large number of correspondences.

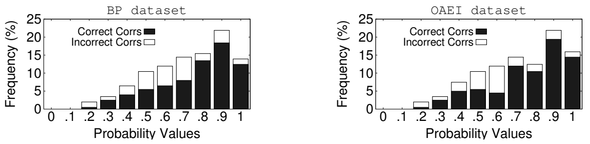

Fig. 6 Relation between probability and correctness of correspondences

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 first compare the candidate correspondences with the exact matching to categorize them as correct or incorrect. Then, we compute the probability of each correspondence. Figure 6 presents a histogram of the probability distribution in the BP dataset (representative for schema matching, other datasets have similar results) and the OAEI dataset for correct and incorrect correspondences. Here, the X-axis depicts the probability ranges and the Y-axis measures the frequency in percentages.

We observe that the probability distribution of correspondences is matched well with their correctness. For example, in the BP dataset, 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 the OAEI dataset, for instance, in the $[0.8,0.9]$ range, there are about 19% correct and about 2% incorrect correspondences; whereas the ratio is about 14%/1% in the $[0.9,1.0]$ range. This indicates that the probability values indeed reflect the correctness of correspondences.

8.3 Evaluation of the Uncertainty Reduction

In this set of experiments, we study to which extent our approach reduces network uncertainty. For each dataset, we generate a complete interaction graph and obtain candidate correspondences using automatic matchers. Then, we simulate the pay-as-you-go reconciliation process where expert input is generated using the exact matches, which had been constructed by the dataset provider. The number of correspondences for each expert interaction is set to $k = 10$.

User guiding strategies. We explored how the quality of the match result in terms of precision improved when eliciting expert input according to different strategies. The BP and OAEI datasets are used for schema matching and ontology alignment, respectively. Figure 7 and 8 depict the improvements in precision and network uncertainty (Y-axes) with increased feedback percentage (X-axis, out of the total number of correspondences) using two strategies, namely

(1) *Rand:* reconciliation using random selection of correspondences, which acts as a baseline for our experiment.
(2) *Heuristic:* we select correspondences using our method that exploits the information-gain (Section 6).

The results depicted in Figure 7 and 8 show the average over 50 experiment runs. The other datasets demonstrate similar results and are omitted for the sake of brevity. We observe a significant reduction of expert effort for our strategy with respect to the baseline. More precisely, applying our solution when selecting correspondences requires only about 50% or less of the expert interactions. Another key finding is that the trends of network uncertainty and precision are inversely similar. This implies that network uncertainty is a good indicator for reconciling the matching results. Note that when network uncertainty is zero (i.e. all integrity constraints are satisfied with a probability of one), the precision is not necessarily guaranteed to be 1.0.

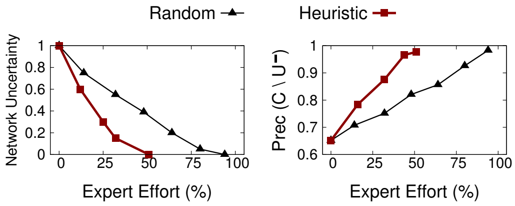

Fig. 7 User effort needed during the reconciliation (BP dataset)

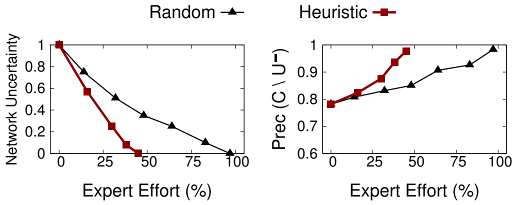

Fig. 8 User effort needed during the reconciliation (OAEI dataset)

Effects of the network topology. We have analyzed the influence of the topology of the interaction graph on the reduction of the required expert input. For this purpose, we used randomly generated interaction graphs, instead of the complete graphs of the previous experiments. We constructed these graphs $G(|\mathscr{S}|,p)$ using the Erdős-Rényi random graph model [31], where p is the inclusion probability that determines whether an edge is included in a graph. We have constructed 10 graphs, and applied the reconciliation procedure. The results are averaged over 5 runs per graph.

Figure 9 (for the BP dataset and the OAEI dataset) depicts the improvements in terms of required expert input, for different graphs. The X-axis corresponds to the inclusion probability used to construct the interaction graphs, while the Y-axis shows the expert efforts. One can observe that our technique significantly reduces the required effort, independently of the topology of the interaction graph. Also these methods are robust w.r.t. the structure of the graph. Moreover, we achieved greater effort reduction in cases where the interaction graph was more dense. This is explained by the richer information on mutual reinforcing dependencies between the random variables in the factor graph representation.

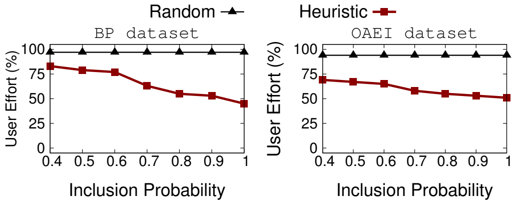

Fig. 9 Effect of network topology

Effects of user knowledge limitation. So far, we assumed that it is always possible to elicit expert input for a correspondence. One may argue that in many practical scenarios, however, this assumption does not hold true. Experts may have only partial knowledge of a domain, which means that for some correspondences feedback cannot be obtained. We studied the performance of our approach in this setting, by including the possibility of skipping a correspondence in the reconciliation process. Thus, for certain correspondences, we never elicit any feedback. However, the probabilistic model allows us to conclude on the assertions for these correspondences as consequences of available expert input. Correspondences that are assigned a probability larger than 0.5 are considered as being correct, whereas the remaining ones are not included in the resulting matching.

Table 3 Ability to conclude assertions

Dataset	p : skipping probability					
	5%	10%	15%	20%	25%	30%
BP	0.89	0.86	0.87	0.83	0.80	0.78
PO	0.81	0.80	0.76	0.72	0.72	0.66
UAF	0.81	0.80	0.76	0.75	0.74	0.71
WebForm	0.71	0.72	0.76	0.79	0.76	0.70
OAEI	0.78	0.77	0.79	0.73	0.72	0.69

In our experiments, we used a probability p for skipping a correspondence and measured the ratio of true assertions (related to skipped correspondences that are concluded correctly against ground truth) and all skipped correspondences. Table 3 shows the obtained results. Even with $p = 30\%$, the ratio is about 0.7, which means that about 70% of the assertions that could not be elicited from the expert have been correctly classified based on the information propagation in factor graph. As expected, this ratio increases as p decreases; skipping less correspondences provides the information propagation with more useful information.

8.4 Evaluation of the Matching Instantiation

Finally, we study the effectiveness of our method for instantiating a matching from a probabilistic matching network.

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

Figure 10 and 11 illustrate the influence of the ordering strategies on quality of the instantiated matching for the BP dataset and the OAEI dataset respectively (again, the other datasets showed the same trend). We varied the amount of expert input (x-axis), considering validation of 0% to 15% of the candidate correspondences. A key finding is that our heuristic ordering strategy generally outperforms the baseline in terms of both precision and recall. Note that initially, with 0% expert input, there is no difference between two ordering strategies since none of the correspondences has been validated. We conclude that our approach to ordering the correspondences to seek expert input plays an important role in improving the quality of the instantiated matching.

Effects of Maximal Likelihood. Instantiation is guided by the size and the likelihood of a particular matching, see Section 7 and we argued that the size shall be maximal to keep us much information on correspondences as possible in the instantiated matching. Yet, in this experiment, we study the importance of also considering the likelihood of correspondences 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 instantiated matching.

Figure 12 and 13 illustrate the percentage of expert input relative to the precision and recall of the instantiated matching for the BP dataset and OAEI dataset, respectively. We observe that considering the likelihood criterion indeed leads to a matching of better quality. The results underline the benefits of our probabilistic model in quantifying the uncertainty of correspondences as well as of the network as a whole.

9 Related Work

We now review the work in pair-wise matching of conceptual models, matching networks, and user feedback that is close to our research.

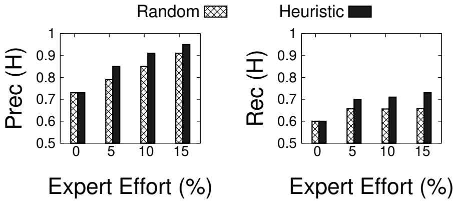

Fig. 10 Effects of correspondence ordering strategies on instantiation. H is the matching instantiated by our algorithm (BP dataset)

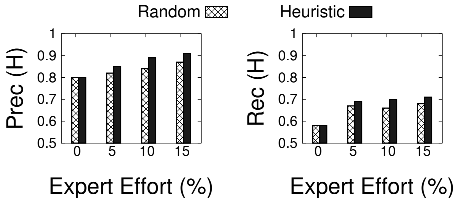

Fig. 11 Effects of correspondence ordering strategies on instantiation. H is the matching instantiated by our algorithm (OAEI dataset)

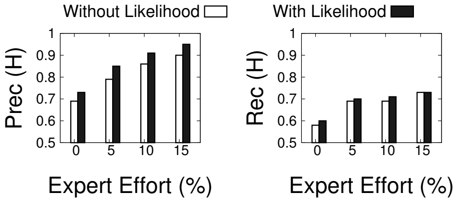

Fig. 12 Effects of the likelihood function on instantiation. H is the matching instantiated by our algorithm (BP dataset)

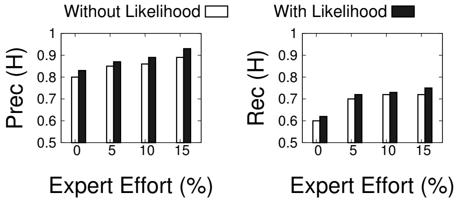

Fig. 13 Effects of the likelihood function on instantiation. H is the matching instantiated by our algorithm (OAEI dataset)

Pair-wise Matching of Conceptual Models. Matching of database schemas is an active research field. The developments of this area have been summarized in several surveys, e.g., [72,13,11,62]. Existing works focused mainly on improving the quality parameters of matchers, such as precision or recall of the generated correspondences [36,93,94,63], or leveraging matching results for further management operations [66,82,83,90]. Recently, however, it was realized 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 [28], systematic matching ensemble selection [37,84,86] or automatic tuning of the matcher parameters [52,36,30,47,71].

Similar research trends and results are also found for other types of conceptual models, such as ontologies [67, 85] or process models [15,7]. Ontology matching focuses on leveraging semantic information such as taxonomies, domain vocabularies, and resource descriptions, to improve matching techniques proposed for database schemas, which lack such semantic information [68]. Process model matching, in turn, also exploits the structure and execution semantics of formal process models, matching on their elements based on a combination of textual, structural, and semantical features [22,21, 12]. Despite having different techniques, existing works on conceptual models only consider pair-wise matching once at a time, thus neglecting a significant amount of network-wise information.

In this paper, we use results from matching tools as input for our approach. Uncertain matching for conceptual models has been studied in many works, see, for instance, [25,76, 35,75,41]. Yet, our approach is the first to consider integrity constraints defined for a matching network to assess the correctness probabilities of correspondences and guide the reconciliation by an expert user.

Matching Networks. The idea of exploiting a set of conceptual models as a whole to improve the matching has been studied before. Holistic matching [81] attempted to exploit statistical co-occurrences of attributes in different database schemas and use them to derive complex correspondences. Corpus-based matching [54] uses a ‘corpus’ of schemas to augment the evidence that improves existing matching and exploit constraints between attributes by applying statistical techniques. Nevertheless, these works follow a mediated approach, which constructs one conceptual model as a single-point reference for all original models. The mediated approach has two limitations: it could be impossible to develop a consensus model that captures all different characteristics of original models and it is difficult to update the mediated model when the original ones are changed. Our matching network approach is applicable for large-scale contexts, where the computation with a monolithic, mediated schema is too costly or simply infeasible.

Network-level constraints were originally considered in [5, 20], in which the establishment of semantic interoperability in large-scale P2P networks was studied. In this paper, we study such integrity constraints in the matching problem of conceptual models and use constraint violations as evidence of matching uncertainty.

User Feedback. The post-matching reconciliation process has also received considerable attention in the literature for database schemas, ontologies, and process models alike. The systems in [48,77,88,27,19,50] rely on one user only, whereas the frameworks in [95,56,63,18,10,60] 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.

The idea of pay-as-you-go approaches to improve the matching quality has been brought forward in [76,44,91, 17]. However, we note that the approach in [76] requires the creation of a mediated model, whereas we study reconciliation for a network of conceptual models. Moreover, unlike many existing works, e.g., [77,88,89,16] that incorporate user feedback implicitly through keywords, our approach lets experts give input explicitly on the correspondences. This results in a clear quantification of expert effort.

10 Conclusions and Future Work

This paper presented an approach to pay-as-you-go reconciliation in matching networks of conceptual models. We defined the notion of a matching network and its representation as a factor graph as the backbone of the approach. Most importantly, our probabilistic graphical model enables us to capture of the uncertainty in the matching network in a unified way, integrating the output of automatic matching, expert feedback, and the model management tasks that shall be solved. Our approach involves three elementary steps that realize a pay-as-you-go setup: establishing the network uncertainty by computing correctness probabilities for correspondences; reducing the network uncertainty with expert input; and instantiating a trusted set of consistent correspondences. As such, the approach can be used for supporting model management 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 formulation can be extended to further develop the quality measurement of matching networks. Second, although the proposed pay-as-you-go approach can be already applied to many model matching tasks (schema matching, ontology alignment, process model matching), more applications which could be transformed into a matching network shall be devised. On top of matching networks, we can develop a wide range of potential utilities in data management systems. Examples of such utilities include visualizing a matching network at large scales, searching & filtering network-level information and data sources efficiently, and reconciling the network dynamically when a new data source arrives.

Acknowledgements 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. `https://one.ubuntu.com`
3. `https://www.dropbox.com`
4. `http://www.eyeos.com`
5. Aberer, K., Cudré-Mauroux, P., Hauswirth, M.: Start making sense: The Chatty Web approach for global semantic agreements. JWS pp. 89–114 (2003)
6. Aberer, K., Cudre-Mauroux, P., Ouksel, A.M., Catarci, T., Hacid, M.S., Illarramendi, A., Kashyap, V., Mecella, M., Mena, E., Neuhold, E.J., Troyer, O.D., Risse, T., Scannapieco, M., Saltor, F., Santis, L.D., Spaccapietra, S., Staab, S., Studer, R.: Emergent semantics principles and issues. In: DASFAA, pp. 25–38 (2004)
7. Antunes, G., Bakhshandeh, M., Borbinha, J.L., Cardoso, J., Dadashnia, S., Francescomarino, C.D., Dragoni, M., Fettke, P., Gal, A., Ghidini, C., Hake, P., Khiat, A., Klinkmüller, C., Kuss, E., Leopold, H., Loos, P., Meilicke, C., Niesen, T., Pesquita, C., Péus, T., Schoknecht, A., Sheetrit, E., Sonntag, A., Stuckenschmidt, H., Thaler, T., Weber, I., Weidlich, M.: The process model matching contest 2015. In: EMISA, pp. 127–155 (2015)
8. Ardissono, L., Goy, A., Petrone, G., Segnan, M.: From service clouds to user-centric personal clouds. In: CLOUD, pp. 1–8 (2009)
9. Aumueller, D., Do, H.H., Massmann, S., Rahm, E.: Schema and ontology matching with coma++. In: SIGMOD, pp. 906–908 (2005)
10. Belhajjame, K., Paton, N.W., Fernandes, A.A.A., Hedeler, C., Embury, S.M.: User feedback as a first class citizen in information integration systems. In: CIDR, pp. 175–183 (2011)
11. Bellahsene, Z., Bonifati, A., Rahm, E.: Schema Matching and Mapping. Springer (2011)
12. Benatallah, B., Sakr, S., Grigori, D., Motahari-Nezhad, H.R., Barukh, M.C., Gater, A., Ryu, S.H., et al.: Process Analytics: Concepts and Techniques for Querying and Analyzing Process Data. Springer (2016)
13. Bernstein, P., Madhavan, J., Rahm, E.: Generic Schema Matching, Ten Years Later. In: VLDB (2011)
14. Branco, M.C., Xiong, Y., Czarnecki, K., Küster, J.M., Völzer, H.: A case study on consistency management of business and IT process models in banking. SoSyM pp. 913–940 (2014)
15. Cayoglu, U., Dijkman, R., Dumas, M., Fettke, P., García-Bañuelos, L., Hake, P., Klinkmüller, C., Leopold, H., Ludwig, A., Loos, P., et al.: Report: The process model matching contest 2013. In: BPM, pp. 442–463 (2014)
16. Chen, H., Yin, H., Wang, W., Wang, H., Nguyen, Q.V.H., Li, X.: Pme: projected metric embedding on heterogeneous networks for link prediction. In: KDD, pp. 1177–1186 (2018)
17. Cong, P.T., Toan, N.T., Hung, N.Q.V., Stantic, B.: Minimizing efforts in reconciling participatory sensing data. In: WIMS, p. 49 (2018)
18. Cruz, I.F., Loprete, F., Palmonari, M., Stroe, C., Taheri, A.: Pay-as-you-go multi-user feedback model for ontology matching. In: EKAW, pp. 80–96 (2014)
19. Cruz, I.F., Stroe, C., Palmonari, M.: Interactive user feedback in ontology matching using signature vectors. In: ICDE, pp. 1321–1324 (2012)
20. Cudré-Mauroux, P., Aberer, K., Feher, A.: Probabilistic Message Passing in Peer Data Management Systems. In: ICDE, pp. 41–52 (2006)
21. Dijkman, R.M., Dumas, M., Ouyang, C.: Semantics and analysis of business process models in bpmn. INFSOF pp. 1281–1294 (2008)
22. Dijkman, R.M., La Rosa, M., Reijers, H.A.: Managing large collections of business process models-current techniques and challenges. CII pp. 91–97 (2012)
23. Do, H.H., Rahm, E.: COMA - A System for Flexible Combination of Schema Matching Approaches. In: VLDB, pp. 610–621 (2002)

24. Doan, A., Domingos, P., Halevy, A.Y.: Reconciling schemas of disparate data sources: a machine-learning approach. In: SIGMOD, pp. 509–520 (2001)
25. Dong, X., Halevy, A.Y., Yu, C.: Data integration with uncertainty. In: PVLDB, pp. 687–698 (2007)
26. Dragisic, Z., Eckert, K., Euzenat, J., Ferrara, A., Granada, R., Ivanova, V., Jiménez-Ruiz, E., Kempf, A.O., Lambrix, P., et al.: Results of the ontology alignment evaluation initiative 2014
27. Duan, S., Fokoue, A., Srinivas, K.: One size does not fit all: Customizing ontology alignment using user feedback. In: ISWC, pp. 177–192 (2010)
28. Duchateau, F., Coletta, R., Bellahsene, Z., Miller, R.J.: (not) yet another matcher. In: CIKM, pp. 1537–1540 (2009)
29. Dumas, M., La Rosa, M., Mendling, J., Reijers, H.A.: Fundamentals of business process management. Springer (2013)
30. Duong, C.T., Nguyen, Q.V.H., Wang, S., Stantic, B.: Provenance-based rumor detection. In: ADC, pp. 125–137 (2017)
31. Erdős, P., Rényi, A.: On the evolution of random graphs. In: Publ. Math. Inst. Hungar. Acad. Sci, pp. 17–61 (1960)
32. Faria, D., Pesquita, C., Santos, E., Cruz, I.F., Couto, F.M.: Agreementmakerlight: a scalable automated ontology matching system. In: DILS, p. 29 (2014)
33. Fensel, D., Lausen, H., Polleres, A., de Bruijn, J., Stollberg, M., Roman, D., Domingue, J.: Enabling semantic web services: the web service modeling ontology. Springer (2006)
34. Franklin, M., Halevy, A., Maier, D.: From databases to dataspaces: a new abstraction for information management. In: SIGMOD, pp. 27–33 (2005)
35. Gal, A.: Uncertain Schema Matching. Morgan & Claypool (2011)
36. Gal, A., Roitman, H., Sagi, T.: From diversity-based prediction to better ontology & schema matching. In: WWW, pp. 1145–1155 (2016)
37. Gal, A., Sagi, T.: Tuning the ensemble selection process of schema matchers. JIS pp. 845–859 (2010)
38. Gal, A., Sagi, T., Weidlich, M., Levy, E., Shafran, V., Miklós, Z., Hung, N.: Making sense of top-k matchings: A unified match graph for schema matching. In: IIWeb (2012)
39. Glover, F., McMillan, C.: The general employee scheduling problem. an integration of ms and ai. COR pp. 563–573 (1986)
40. Goldberg, D.E.: Genetic Algorithms in Search, Optimization and Machine Learning. Addison-Wesley Longman (1989)
41. Gong, J., Cheng, R., Cheung, D.W.: Efficient management of uncertainty in XML schema matching. JVLDB pp. 385–409 (2012)
42. Gonzalez, H., Halevy, A.Y., Jensen, C.S., Langen, A., Madhavan, J., Shapley, R., Shen, W., Goldberg-Kidon, J.: Google fusion tables: web-centered data management and collaboration. In: SIGMOD, pp. 1061–1066 (2010)
43. Hopcroft, J., Tarjan, R.: Algorithm 447: efficient algorithms for graph manipulation. CACM pp. 372–378 (1973)
44. Hung, N.Q.V., Duong, C.T., Tam, N.T., Weidlich, M., Aberer, K., Yin, H., Zhou, X.: Argument discovery via crowdsourcing. VLDB J. pp. 511–535 (2017)
45. Hung, N.Q.V., Tam, N.T., Chau, V.T., Wijaya, T.K., Miklós, Z., Aberer, K., Gal, A., Weidlich, M.: SMART: A tool for analyzing and reconciling schema matching networks. In: ICDE, pp. 1488–1491 (2015)
46. Hung, N.Q.V., Tam, N.T., Miklós, Z., Aberer, K.: Reconciling schema matching networks through crowdsourcing. EAI p. e2 (2014)
47. Hung, N.Q.V., Zheng, K., Weidlich, M., Zheng, B., Yin, H., Tam, N.T., Stantic, B.: What-if analysis with conflicting goals: Recommending data ranges for exploration. In: ICDE, pp. 1–12 (2018)
48. Jeffery, S.R., Franklin, M.J., Halevy, A.Y.: Pay-as-you-go user feedback for dataspace systems. In: SIGMOD, pp. 847–860 (2008)
49. Karp, R.M.: Reducibility Among Combinatorial Problems. In: COCO, pp. 85–103 (1972)
50. Klinkmüller, C., Leopold, H., Weber, I., Mendling, J., Ludwig, A.: Listen to me: Improving process model matching through user feedback. In: BPM, pp. 84–100 (2014)
51. Kschischang, F.R., Frey, B.J., Loeliger, H.A.: Factor graphs and the sum-product algorithm. IEEE Trans. Inf. Theory pp. 498–519 (2001)
52. Lee, Y., Sayyadian, M., Doan, A., Rosenthal, A.S.: eTuner: tuning schema matching software using synthetic scenarios. JVLDB pp. 97–122 (2007)
53. Litwin, W., Mark, L., Roussopoulos, N.: Interoperability of multiple autonomous databases. CSUR pp. 267–293 (1990)
54. Madhavan, J., Bernstein, P.A., Doan, A., Halevy, A.: Corpus-based schema matching. In: ICDE, pp. 57–68 (2005)
55. Marthi, B., Pasula, H., Russell, S., Peres, Y.: Decayed mcmc iltering. In: UAI, pp. 319–326 (2002)
56. McCann, R., Shen, W., Doan, A.: Matching Schemas in Online Communities: A Web 2.0 Approach. In: ICDE, pp. 110–119 (2008)
57. Nemhauser, G., Wolsey, L., Fisher, M.: An analysis of approximations for maximizing submodular set functions–i. MP pp. 265–294 (1978)
58. Nemhauser, G.L., Wolsey, L.A.: Maximizing submodular set functions: formulations and analysis of algorithms. North-Holland Mathematics Studies pp. 279–301 (1981)
59. Ngo, D.H., Bellahsene, Z.: YAM++ : (not) Yet Another Matcher for Ontology Matching Task. In: BDA (2012)
60. Nguyen, Q.V., Luong, H.X., Miklós, Z., Aberer, K.: Collaborative schema matching reconciliation. In: CoopIS (2013)
61. Nguyen, Q.V.H.: Reconciling schema matching networks. Ph.D. thesis, EPFL (2014)
62. Nguyen, Q.V.H., Do, S.T., Nguyen Thanh, T., Aberer, K.: Towards enabling schema reuse with privacy constraints. Tech. rep., EPFL (2013)
63. Nguyen, Q.V.H., Nguyen, T.T., Miklós, Z., Aberer, K.: On leveraging crowdsourcing techniques for schema matching networks. In: DASFAA, pp. 139–154 (2013)
64. Nguyen, Q.V.H., Nguyen, T.T., Miklós, Z., Aberer, K., Gal, A., Weidlich, M.: Pay-as-you-go reconciliation in schema matching networks. In: ICDE, pp. 220–231 (2014)
65. Nguyen, Q.V.H., Wijaya, T.K., Miklos, Z., Aberer, K., Levy, E., Shafran, V., Gal, A., Weidlich, M.: Minimizing Human Effort in Reconciling Match Networks. In: ER (2013)
66. Nguyen, T.T., Nguyen, Q.V.H., Weidlich, M., Aberer, K.: Result selection and summarization for web table search. In: ICDE, pp. 231–242 (2015)
67. Noy, N.F., Musen, M.A.: Algorithm and tool for automated ontology merging and alignment. In: AAAI (2000)
68. Otero-Cerdeira, L., Rodríguez-Martínez, F.J., Gómez-Rodríguez, A.: Ontology matching: A literature review. ESWA pp. 949–971 (2015)
69. Paolucci, M., Kawamura, T., Payne, T.R., Sycara, K.: Semantic matching of web services capabilities. In: ISWC, pp. 333–347 (2002)
70. Peukert, E., Eberius, J., Rahm, E.: AMC - A framework for modelling and comparing matching systems as matching processes. In: ICDE, pp. 1304–1307 (2011)
71. Qian, T., Liu, B., Nguyen, Q.V.H., Yin, H.: Spatiotemporal representation learning for translation-based poi recommendation. TOIS **37**(2), 18 (2019)
72. Rahm, E., Bernstein, P.A.: A Survey of Approaches to Automatic Schema Matching. JVLDB pp. 334–350 (2001)
73. Roitman, H., Gal, A.: Ontobuilder: fully automatic extraction and consolidation of ontologies from web sources using sequence semantics. In: EDBT, pp. 573–576 (2006)
74. Russell, S.J., Norvig, P., Canny, J.F., Malik, J.M., Edwards, D.D.: Artificial intelligence: a modern approach. Prentice hall Englewood Cliffs (1995)

75. Sarasua, C., Simperl, E., Noy, N.F.: Crowdmap: Crowdsourcing ontology alignment with microtasks. In: ISWC, pp. 525–541 (2012)
76. Sarma, A.D., Dong, X., Halevy, A.Y.: Bootstrapping pay-as-you-go data integration systems. In: SIGMOD, pp. 861–874 (2008)
77. Sayyadian, M., LeKhac, H., Doan, A., Gravano, L.: Efficient keyword search across heterogeneous relational databases. In: ICDE, pp. 346–355 (2007)
78. Seligman, L., Mork, P., Halevy, A., Smith, K., Carey, M.J., Chen, K., Wolf, C., Madhavan, J., Kannan, A., Burdick, D.: Openii: an open source information integration toolkit. In: SIGMOD, pp. 1057–1060 (2010)
79. Shannon, C.E., Weaver, W.: A mathematical theory of communication (1948)
80. Smith, K., Morse, M., Mork, P., Li, M., Rosenthal, A., Allen, D., Seligman, L., Wolf, C.: The role of schema matching in large enterprises. In: CIDR (2009)
81. Su, W., Wang, J., Lochovsky, F.: Holistic schema matching for web query interfaces. In: EDBT, pp. 77–94 (2006)
82. Tam, N.T., Hung, N.Q.V., Quan, T.T.: A framework to combine multiple matchers for pair-wise schema matching. In: RIVF, pp. 1–6 (2012)
83. Toan, N.T., Cong, P.T., Thang, D.C., Hung, N.Q.V., Stantic, B.: Bootstrapping uncertainty in schema covering. In: ADC, pp. 336–342 (2018)
84. Tran, L.H., Nguyen, Q.V.H., Do, N.H., Yan, Z.: Robust and hierarchical stop discovery in sparse and diverse trajectories. Tech. rep., EPFL (2011)
85. Vargas-Vera, M., Nagy, M.: State of the art on ontology alignment. IJKSR pp. 17–42 (2015)
86. Wang, W., Yin, H., Huang, Z., Wang, Q., Du, X., Nguyen, Q.V.H.: Streaming ranking based recommender systems. In: SIGIR, pp. 525–534 (2018)
87. Weidlich, M., Mendling, J., Weske, M.: A foundational approach for managing process variability. In: CAISE, pp. 267–282 (2011)
88. Yan, Z., Zheng, N., Ives, Z.G., Talukdar, P.P., Yu, C.: Actively soliciting feedback for query answers in keyword search-based data integration. In: PVLDB, pp. 205–216 (2013)
89. Yin, H., Chen, L., Wang, W., Du, X., Hung, N.Q.V., Zhou, X.: Mobi-sage: A sparse additive generative model for mobile app recommendation. In: ICDE, pp. 75–78 (2017)
90. Yin, H., Hu, Z., Zhou, X., Wang, H., Zheng, K., Hung, N.Q.V., Sadiq, S.W.: Discovering interpretable geo-social communities for user behavior prediction. In: ICDE, pp. 942–953 (2016)
91. Yin, H., Zhou, X., Cui, B., Wang, H., Zheng, K., Hung, N.Q.V.: Adapting to user interest drift for POI recommendation. TKDE pp. 2566–2581 (2016)
92. Zhang, C., Ré, C.: Towards high-throughput gibbs sampling: A study across storage managers. In: SIGMOD, pp. 397–408 (2013)
93. Zhang, C.J., Chen, L., Jagadish, H., Cao, C.C.: Reducing uncertainty of schema matching via crowdsourcing. In: VLDB, pp. 757–768 (2013)
94. Zhang, C.J., Zhao, Z., Chen, L., Jagadish, H.V., Cao, C.C.: Crowdmatcher: crowd-assisted schema matching. In: SIGMOD, pp. 721–724 (2014)
95. Zhdanova, A.V., Shvaiko, P.: Community-Driven Ontology Matching. In: ESWC, pp. 34–49 (2006)

A Learning Constraint Parameter

Under our probabilistic model, each constraint $\gamma \in \Gamma$ is associated with a parameter Δ, as illustrated above with the 1-to-1 constraint, the cycle constraint, and the structure constraint. In practice, the parameter Δ is often specified by the application expert or administrator. However, as reconciliation is an incremental process, in this work we propose an adaptive learning method to adjust Δ based on the user answers obtained so far.

More precisely, we use the following heuristic to learn the parameter Δ for each constraint γ. The idea is that the more violations the user make, the more the associated constraints should be hardened; and vice-versa. Initially, we set $\Delta = 0.5$ since the integrity constraints do not affect the correctness of validated correspondences. Then periodically (e.g. after obtaining other 20 answers from the user), we compute the set of constraint violations on top of all correspondences he approved. Denote $V = \{v_1, \ldots, v_n\}$ as the union set of all constraint violations (note that two different violations can be of the same constraint). For each violation $v_i \in V$, we count the number of correspondences in this violation. Then for each constraint γ involved in V, we set its new parameter Δ to the average value of the percentages of its violations.