EAI Endorsed Transactions
on Collaborative Computing

Research Article ICST.ORG

Reconciling Schema Matching Networks Through Crowdsourcing

Nguyen Quoc Viet Hung[1], Nguyen Thanh Tam[1], Zoltán Miklós[2], Karl Aberer[1]

[1]École Polytechnique Fédérale de Lausanne
[2]Université de Rennes 1

Abstract

Schema matching is the process of establishing correspondences between the attributes of database schemas for data integration purposes. Although several automatic schema matching tools have been developed, their results are often incomplete or erroneous. To obtain a correct set of correspondences, usually human effort is required to validate the generated correspondences. This validation process is often costly, as it is performed by highly skilled experts. Our paper analyzes how to leverage crowdsourcing techniques to validate the generated correspondences by a large group of non-experts.

In our work we assume that one needs to establish attribute correspondences not only between two schemas but in a network. We also assume that the matching is realized in a pairwise fashion, in the presence of consistency expectations about the network of attribute correspondences. We demonstrate that formulating these expectations in the form of integrity constraints can improve the process of reconciliation. As in the case of crowdsourcing the user's input is unreliable, we need specific aggregation techniques to obtain good quality. We demonstrate that consistency constraints can not only improve the quality of aggregated answers, but they also enable us to more reliably estimate the quality answers of individual workers and detect spammers. Moreover, these constraints also enable to minimize the necessary human effort needed, for the same expected quality of results.

Received on 11 July 2014, accepted on 23 July 2014, published on 20 October 2014

Keywords: data integration, schema matching, crowdsourcing, worker assessment, user effort

doi: 10.4108/ cc.1.2.e2

1. Introduction

More and more online services enable users to upload and share structured data, including Google Fusion Tables [1], Freebase [2], and Factual [3]. These services primarily offer easy visualization of uploaded data as well as tools to embed the visualization to blogs or Web pages. As fragmentation of data in different sources is a common phenomenon, it is essential to create the interlinks between them [4]. 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 [1]. Extraction of information over all regions requires means to query and aggregate across multiple tables, thereby raising the need of interconnecting schemas to achieve an integrated view of the data. The number of publicly available datasets grows rapidly, making the integration more and more challenging.

In all of the above contexts one needs to integrate data that is stored using different schemas and the interactions between the datasets (or schemas) form a network. Designing a mediated schema for the entire network of schemas might be impractical, especially if the number of schemas that are involved in a network changes, as this might require modifications also in the mediated schema. For this reason we study schema matching techniques that construct the attribute correspondences in a pairwise fashion between the schemas of the network. In this case, we need to ensure that the created correspondences are globally (and not only pairwise) consistent in the network, since these natural consistency conditions are important for the applications.

Since automatic schema matching tools rely on heuristic techniques [5, 6], their result is inherently uncertain. In practice, data integration tasks frequently include a post-matching reconciliation, in which correspondences are reviewed and validated by a human expert. This post-matching reconciliation phase is often the most costly part of schema matching (or even data integration), because of the involvement of expensive experts. In our previous work [7] we demonstrated that the network-level integrity constraints can be exploited to reduce this effort.

In the current paper we analyze how to realize this task through crowdsourcing, where not a single expert but a group of non-experts reconcile the network of attribute correspondences. In this way we leverageing the "wisdom of the crowd" to assert correspondences. Crowdsourcing is a promising approach to reduce the costs of involvement of human experts. While one could potentially reduce the costs, we need to adopt specific techniques to deal with the quality problems that might be present in the user input. This paper is a largely extended version of our own paper [8], where we already demonstrate the use of crowdsourcing techniques for schema matching reconciliation. In this paper, we consolidate this approach by showing that the integrity constraints can be also useful for assessing workers, towards improving the reconciliation quality.

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

- Section 2: We provide an overview of our crowdsourcing framework, including the elements of a matching network, reconciliation through crowdsourcing, the probabilistic model for integrity constraints. We also present a system where our techniques and algorithms can be realized that we also used to evaluate our techniques.

- Section 3: We provide a probabilistic model for combining the answers from multiple crowd workers. This model enables to compute for each candidate correspondence the probability whether it is true.

- Section 4: We show how to evaluate and control the worker quality. On the one hand, we would like to take into account that crowd workers have wide-ranging levels of expertise, thus the quality of their responses can also vary. On the other hand we would like to detect spammers, who exploit the platform to obtain payments for completing tasks without a proper engagement. In particular, we propose mechanisms to detect an individual spammer and groups of spammers, since these malicious workers lower the accuracy of the aggregated results.

- Section 5: We design an aggregation mechanism to instantiate the final decision for each correspondence using the computed probabilities. In particular, we study how to aggregate answers in the presence of matching network constraints. Our theoretical and empirical results show that by harnessing the network constraints, the worker effort can be lowered considerably.

- Section 6: We run the experiments on real datasets to show the effectiveness of leveraging integrity constraints in worker assessment and answer aggregation.

The remaining sections are structured as follows. Section 7 summarizes related work, before Section 8 concludes the paper.

2. Overview

This section starts with a motivating example of a network of schemas. Although the involved schemas are simple, they are sufficient to demonstrate certain problems that arise, when we attempt to interconnect their attributes. Next, we explain the elements of a matching network and our techniques for using crowdsourcing to validate the correspondences in this network. Then, we proceed with a formulation of integrity constraints in our probabilistic model. Finally, we describe our framework for obtaining aggregate values for attribute correspondences from potentially unreliable crowd answers.

2.1. Motivating Example

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

Problematic correspondences are typically eliminated by reconciliation based on human input: a given correspondence is asserted whether it shall be disregarded or accepted [9]. Since a large-scale matching network has a lot of correspondences, reconciling through human experts could be expensive, in fact this is usually the most costly phase of schema matching, as it involves human efforts. One can hope to reduce this cost by using a crowdsourcing platform, where a group of non-experts can execute this task. If we let execute the reconciliation task by crowd workers, we need to take attention to various issues. In particular, we need to cope with incorrect user input that might

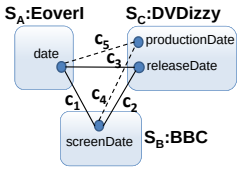

Figure 1. A matching network of real-world schemas

even come from malicious users (for example, from those who just want to obtain the financial compensation for the task without providing useful input). While asking more questions, thus increasing the tasks assigned to the crowd, could improve the quality of the (aggregated) results, this would at the same time increase the financial costs, that we would minimize. As we will demonstrate the use of network-level integrity constraints can help both with quality improvements and also with cost reduction: these constraints can on the one hand help to reduce the number of necessary questions we need to ask, this it can reduce the overall cost of work, and at the same time, integrity constraints can largely contribute to detect quality problems with use input.

We illustrate these effect in Figure 1. We can see that if we approve both c_3, c_5, the two attributes productionDate and releaseDate are equivalent, which is intuitively incorrect. Regarding the malicious worker problem, we can approach as follows. The workers who approve both these correspondences should be penalized by decreasing their reliability. Based on this evidence, we can assess the reliability of workers and, for instance, filter the low-quality workers out of the pool. Regarding the cost minimization problem, we can approach as follows. We can see that if c_3 is approved by all workers, there is no need to validate c_5 since c_5 and c_3 cannot be true at the same time. This leads to a smaller number of necessary questions; and thus, reduces the monetary cost for paying the workers.

In the following we introduce the schema matching network model [7] that we we will use in our work.

2.2. Matching Networks and Reconciliation through Crowdsourcing

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

An *attribute correspondence* between a pair of schemas $s_1, s_2 \in \mathcal{S}$ is an attribute pair (a, b), such that $a \in s_1$ and $b \in s_2$. The set of *candidate correspondences* $C_{i,j}$ for a pair of schemas $s_i, s_j \in \mathcal{S}$ is a set of attribute correspondences which is typically the outcome of schema matchers [10]. The set of candidate correspondences C for an interaction graph $G_{\mathcal{S}}$ consists of all candidate correspondences for pairs of schemas corresponding to its edges, i.e. $C = \bigcup_{(s_i,s_j)\in E(G_{\mathcal{S}})} C_{i,j}$. Although more complex models for correspondences have been proposed, cf., [11], we focus on correspondences modelled as attribute pairs since this model is followed by the majority of schema matchers [5, 6].

Based on the above notions, we define a network of schemas to be a triple $N = \langle \mathcal{S}, G_{\mathcal{S}}, C \rangle$, where $\mathcal{S}$ is a set of schemas (of unique attributes), $G_{\mathcal{S}}$ is an interaction graph, Γ is a set of constraints, and C is a set of candidate correspondences. Since the results of automatic matchers are inherently uncertain, the set of candidate correspondences C of N does not often provide a satisfactory result for a data integration task. Instead, we are interested in finding the ground truth – the set of all correspondences that are correct. However, the ground truth is hidden and could not be known before-hand. We denote X_c as a random variable of the existence of a correspondence c in the ground truth. X_c being equal to $true/false$ indicates that c exists/not exists in ground truth. We try to estimate the value of X_c with the help of crowd workers.

Reconciliation of a network of schemas $N = \langle \mathcal{S}, G_{\mathcal{S}}, C \rangle$ through crowdsourcing is an incremental process, where a set of workers $W = \{w_1, \ldots, w_k\}$ provide input to the answer matrix $[M_{ij}]_{|C|\times|W|}$. Each element M_{ij} is the validation of worker w_j on the correspondence $c_i \in C$. Domain values of M_{ij} are $\{true, false, null\}$, where $true/false/null$ indicates c_i is approved/disapproved/not validated yet. These validation results correspond to simple validation tasks that are proposed to workers. Note that we might ask the same question from several workers, while each worker receives a particular question only once. Since crowd workers have wide-ranging levels of expertise, they provide different answers for a correspondence. To aggregate the worker's answers, we compute for each correspondence c the probability $Pr(X_c = true)$ that c exists in ground truth. The set of all these probabilities of candidate correspondences is denoted as $P = \{Pr(X_c = true) \mid c \in C\}$. Summing up the above notions, we denote the state of the crowdsourced matching network as a tuple $\langle N, W, M, P \rangle$, where N is the network of schemas, W is the worker pool, M is the answer matrix, and P is the probability set.

2.3. Integrity Constraints

We can express natural expectations that one has w.r.t. the entire network in the form of consistency

constraints as follows. Given a network of schemas $N = \langle \mathcal{S}, G_{\mathcal{S}}, C \rangle$, let us denote $\Gamma = \{\gamma_1, \ldots, \gamma_n\}$ be a finite set of constraints that are used to represent the expected consistency conditions on N. We say that a set of correspondences $C' \in C$ violating a constraint $\gamma \in \Gamma$ is a constraint violation. In practice, we are not interested in all possible violations, but the minimal ones: We say that a violation is minimal w.r.t. γ, if none of its proper subsets is violating γ.

In [7], we relied on Answer Set Programming formalism to express the integrity constraints. In this paper, we provide a different formulation for the same constraints, using probabilities. The advantage of this formulation is that the constraints can be softened and parametrized based on particular scenarios. We do not impose assumptions on the definition of integrity constraints. For illustration, we rely on the examples of the one-to-one constraint and cycle constraint as defined in [7].

Generalized 1-1 constraint. Each attribute of one schema should be matched to at most one attribute of any other schema. For example in Figure 1, the set $\{c_3, c_5\}$ violates the 1-1 constraint. However there are some exceptions where this constraint does not hold, such as the attribute *name* of a schema might be a concatenation of the attributes $firstname$ and $lastname$ of another schema. To capture this observation, we provide a relaxed version of the constraint using probability theory:

$$Pr(\gamma_{1-1}|X_{c_0}, X_{c_1}, \ldots, X_{c_k}) = \begin{cases} 1 & \text{If } m \leq 1 \\ \Delta \in [0,1] & \text{If } m > 1 \end{cases} \quad (1)$$

where $\{c_0, c_1, \ldots, c_k\}$ is a set of correspondences that share a common source attribute and m is the number of X_{c_i} assigned as *true*. When $\Delta = 0$, there is no constraint exception (the constraint is hard). The constraint can be softened by adjusting the Δ value.

Cycle constraint. If multiple schemas are matched in a cycle, the matched attributes should form a closed cycle. For example in Figure 1, the set $\{c_1, c_2, c_5\}$ violates the *cycle constraint*. Formally, following the notion of cyclic mappings in [12], we formulate the conditional probability of a cycle as follows:

$$Pr(\gamma_{\circlearrowleft}|X_{c_0}, X_{c_1}, \ldots, X_{c_k}) = \begin{cases} 1 & \text{If } m = k+1 \\ 0 & \text{If } m = k \\ \Delta \in [0,1] & \text{If } m < k \end{cases} \quad (2)$$

where $c_0, c_1, \ldots, c_k$ forms a sequence of correspondences that starts and ends at the same attribute; and m is the number of X_{c_i} assigned as $true$ and Δ is the probability of compensating errors along the cycle (i.e., two or more incorrect assignment resulting in a correct reformation).

Learning Constraint Parameter. Under our probabilistic model, each constraint $\gamma \in \Gamma$ is associated with a parameter Δ, as illustrated with the one-to-one constraint and cycle constraint above. In practice, the parameter Δ is often specified by the application expert or administrator. However, as crowdsourcing is an incremental process, in this work we propose an adaptive learning method to adjust Δ based on the worker 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 workers make, the more the associated constraints should be hardened; and vice-versa. Initally, 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 crowd), we compute the set of constraint violations, for each worker, on the set of 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 percentage of workers who made 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.

2.4. Crowdsourcing Framework

Figure 2 presents the overall reconciliation process through crowdsourcing and the global architecture of our platform. The reconciliation process starts with a set of candidate correspondences that are generated by schema matchers. Based on these candidate correspondences, we initialize a network of candidate correspondences. We employ workers from the crowd to answer validation questions, i.e. for each correspondence, we automatically generate a crowd task where we ask a worker to validate the correspondence. The crowd workers then provide their input.

During the process, the workers are continuously evaluated by the component *Worker Assessment*. Since the workers might provide different answers for the same question, we need to estimate the probability that a given correspondence is true. *Probability Computation* component is responsible for these calculations. The component *Answer Aggregation* aggregates the workers answers into a single decision. In the end, the output of our framework is an aggregated matching, which consist of the correspondences, their aggregated values and the associated error rates.

Following this general structure, crowdsourcing for schema matching network requires the realization of the following components.

Probability Computation. Given a crowdsourced matching network, this component computes the

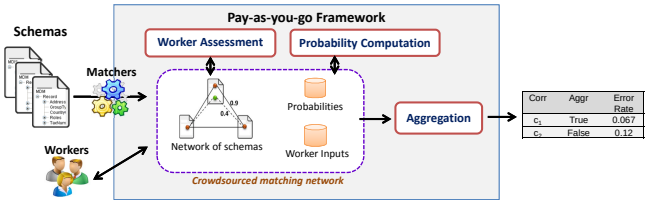

Figure 2. Architecture of the crowdsourced reconciliation framework

probability of each correspondence whether it should be correct, based on the worker inputs obtained so far. More precisely, it computes the probability $Pr(X_c = true)$ of each correspondence $c \in C$. Then, the resulting probabilities can be used for the other components since any worker input is implicitly incorporated in the probabilities. The probability computation is described in Section 3.

Worker Assessment. This component is responsible for evaluating worker reliability based on their answers. In this paper, we tackle the problem of detecting low-quality workers. As soon as the spammers workers are detected, we can remove them out of the worker pool and exclude their answers from the aggregation procedures. This can also save the monetary cost and improve the quality of the aggregated matching. Our worker assessment techniques are described in Section 4.

Answer Aggregation. Given a matching network at the end of a reconciliation process through crowdsourcing, the answer aggregation component will decide the final value of the validated correspondences. More precisely, it decides whether a given correspondence should exist in the final matching and provides an (estimated) error rate of this decision. The details of this component is described in Section 5.

3. Probability Computation

In this Section we discuss the the techniques we used to compute $Pr(X_c)$, the probability that a given correspondence c is true. There are several techniques proposed in the literature to compute this probability [13]. In this paper, we use the expectation-maximization (EM) algorithm, which aggregates all answers of workers and estimates their reliability simultaneously. The reason behind this choice is that the EM model is quite effective for labelling tasks and robust to noisy workers [14].

In the following, we provide a formulation of the Expectation-Maximization (EM) algorithm, which is inspired from [15–17]. The EM algorithm takes as input an answer matrix $[M_{ij}]_{n\times m}$ (n correspondences and m workers) and returns a tuple $\langle P, V\rangle$. V is a vector in which each element v_j is the (estimated) quality of the worker w_j. P is a vector in which each element p_i is the (estimated) probability of correctness for each correspondence c_i. The algorithm alternates between two steps: Expectation step (E-step) and Maximization step (M-step) until it reaches a convergence state where the estimated values of v_j and p_i are stable. In the k-th E-step, it takes the calculated worker quality V^{k-1} estimated in the previous step to calculate the probability of correctness for the correspondences P^k in this step according to the following equation:

$$p_i^k = \sum_{t=1}^{m} v_t^{k-1} \times f(M_{it}) \times \mathbb{1}_{M_{it}=true} \tag{3}$$

where f is a function that estimates the correctness of the answers given by the workers and $\mathbb{1}_{cond} = 1$ if $cond$ is true and 0 otherwise. In practice, we can estimate the value of f by the probability of correctness for the answer M_{ij} calculated in the previous step. After this step, for each correspondence, the correct answer can be estimated by selecting the one with the highest probability. For correspondences that have been validated from workers, we take the provided answers as the correct values. We denote the estimated correct values at step k as $G^k = \{g_1, g_2, ..., g_n\}$ where g_i is the correct answer for correspondence c_i.

Since the estimated correct values change after each E-step, we need to update the estimated quality of the workers to reflect these changes. In the k-th M-step, we re-estimate the quality of the workers by computing the loss value L_i^k for each worker. This loss value measures how deviating the answers provided by a worker to the estimated correct values:

$$L_j^k = \sum_{i=1}^{n} v_j \times h(M_{ij}, g_i) \tag{4}$$

where h is a function that measure the distance between two values. Based on the loss value of each worker, we can re-estimate its quality based on the intuition that the higher the loss value, the lower the quality of the worker.

In the end, the probabilities of possible aggregations of each correspondence c_i are:

$$\begin{cases} Pr(X_{c_i} = true) = p_i \\ Pr(X_{c_i} = false) = 1 - p_i \end{cases} \tag{5}$$

4. Worker Assessment

The potentially low quality of results obtained from crowd workers is a major problem for crowdsourcing applications. While a large group of crowd workers from diverse background can perform challenging tasks, individual workers are typically not experts on any particular domain thus their responses cannot be

trusted. A common way to cope with this situation is to ask the same question multiple times and aggregate the results. There is however often a big difference in the expected quality of answers from different workers. A possible way is to cope with this situation is to assess the quality of the workers and use this assessment when aggregating the answers.

Besides the quality problems that exists with honest crowd workers with varying response quality, there is another problem that can largely influence the quality of results obtained from the crowd. As the workers are paid upon completion of a task, there is a group of workers who just would like to obtain the offered money, by giving random answers to the questions (or by providing answers that are identical to a honest worker). This group is often referred as spammers. In this section we propose techniques to detect spammers.

Spammers exist very frequently in online communities, especially at crowdsourcing platforms. Several experiments [18, 19] have showed that the proportion of spammers at popular crowdsourcing platforms could be up to 40%. In this way they can significantly increase the cost (since they need to be paid) and at the same time decrease the accuracy of final aggregated results, thus a mechanism to detect and eliminate the spammer's responses is desirable for any crowdsourcing application.

In the following, we first show our methods for detecting an individual spammer. Then we demonstrate how to extend these methods to detect a group of spammers.

4.1. Detecting Individual Spammers

There is a simple simple technique for detecting spammers: one needs to prepare a test set where one knows the answers to the questions. If we ask these test questions then we can estimate their performance and consider these estimations as result quality. The workers who fail to answers correctly a specified minimal number of questions are regarded as spammers. This schema has however several disadvantages: since the spammers already know they are tested, they can work honestly to bypass the test. As the test questions are injected implicitly into the question set, we have to pay the workers for the extra questions.

We propose a constraint-based detection mechanism, that is applicable in our context, where we do not need to add test questions. Spammers validate the correspondences randomly thus their input can create constraint violations. We define a *violation rate* (VR) for each worker, calculated by:

$$VR(w_i) = \frac{\mu(C_i)}{\mu(C)} \tag{6}$$

where $C_i = \{c_j \in C \mid M_{ij} = true\}$ is the set of correspondences approved by worker w_i and C is the original set of candidate correspondences. The higher value of VR a worker has, the higher chance that he is a spammer. For the detection, we define a filtering threshold α and regard all workers with $VR > \alpha$ as spammers. In the experiments, we vary this threshold and study its effect.

The $\mu(.)$ function measures the amount of violations for a set of correspondences and is formulated as follows. Given a set of correspondences $C' \in C$, we can obtain a set of violations $Vio_{C'} = \{v_1, \ldots, v_m\}$ following the definition of integrity constraints in Section 2.3. Also due to this definition, each violation v_i is associated with the constraint parameter Δ_i that reflects the chance that the constraint of this violation is incorrect. In other words, $1 - \Delta_i$ reflects the chance that the violation v_i is correct. Formally, we have:

$$\mu(C') = \sum_{v_i \in Vio_{C'}} (1 - \Delta_i) \tag{7}$$

The value domain of $\mu(.)$ is $[0, |Vio_{C'}|]$. Note that if all the integrity constraints are hard constraints (i.e. $\Delta_i = 0, \forall i$), the function $\mu(.)$ becomes the counting function for the number of violations (i.e. $\mu(.) = |Vio_{C'}|$).

The advantage of our detection scheme is that it can be used at any time during the crowdsourcing process to remove the spammers out of the worker pool. The spammers do not know before-hand they are being tested; and thus, avoiding any preparation from them. Moreover, it does not take any extra cost for either paying workers for the test questions or designing the test.

4.2. Detecting Spammer Groups

One of the major problems in crowdsourcing systems is the existence of imitating groups (or spammer groups). Such groups arise in the following way: a user created multiple clone accounts, executes the tasks at his main account and he gives the same answers to questions at all of his clone accounts in order to obtain more money for completed tasks. In the presence of imitating groups, the final aggregated decision would be compromised since false answers may be duplicated and dominate the result. For example, consider five workers w_1, w_2, w_3, w_4, w_5 validating a correspondence whose correct answer is *yes*. Assume w_3, w_4 and w_5 is an imitating group. The answers of five workers are $\{yes, yes, no, no, no\}$ respectively. Without knowing this imitating group, the aggregated answer is *no* (w.r.t. majority voting). Whereas, if we can detect this group and eliminate w_4 and w_5, the aggregated answer is *yes* (w.r.t majority voting). This example shows that the imitating groups are indeed dangerous and we need to design a detection mechanism for this specific scenario.

However, detecting imitating groups is challenging since it is often hard to know whether a worker shares the same answers with another by imitating or by accidence. A naive method to detect the imitation is using the number of correct answers and incorrect answers that two given workers share with each other. The idea is that two workers who share more incorrect answers have a higher chance of belonging to an imitating group. However, this method is not applicable in our setting since the ground truth is unknown (i.e. cannot know for sure an answer is correct or not).

In this section, we propose a detection mechanism based on constraint violations. The idea is that the answers of two workers having the same violations would have high chances of being incorrect since each violation must have at least one incorrect answer. Thus the more violations the two workers share, the high chance these workers belong to an imitating group.

Problem Formulation. Given two workers $w_1, w_2 \in W$ whose answers are all identical, we would like to detect whether these two workers belong to an imitating group (denoted as $w_1 \sim w_2$) or not (denoted as $w_1 \perp w_2$). Denote $V = \{v_1, \ldots, v_k\}$ as the set of identical violations between w_1 and w_2 (assuming that the violations are disjoint). Let $D = \cup_{v_i \in V} v_i$ be the union set of all correspondences in these violations. Denote k_t and k_f as the number of common correct and incorrect answers between the two workers respectively; i.e. $|D| = k_t + k_f$.

To solve the detection problem, we approach by using the probability theory. More precisely, we aim to compute the probability that w_1 and w_2 belong to an imitating group given their common violations:

$$Pr(w_1 \sim w_2|D) = \frac{Pr(D|w_1 \sim w_2) \cdot \alpha}{Pr(D|w_1 \sim w_2) \cdot \alpha + Pr(D|w_1 \perp w_2) \cdot (1-\alpha)} \quad (8)$$

where $\alpha = Pr(w_1 \sim w_2) = 1 - Pr(w_1 \perp w_2)$ $(0 < \alpha < 1)$ is the a-priori probability that there exists an imitating group in the crowd. Now we need to compute the two probabilities $Pr(D|w_1 \sim w_2)$ and $Pr(D|w_1 \perp w_2)$. To this end, we denote r $(0 \leq r \leq 1)$ as the probability that an independently provided answer is true.

We first consider the case where w_1 and w_2 do not belong to an imitating group. Since each correspondence either exists or does not exist in ground truth, the probability that w_1 and w_2 provide the same correct answer for a correspondence $c \in D$ is:

$$Pr(c \text{ is correct}|w_1 \perp w_2) = r \cdot r = r^2$$

And the probability that w_1 and w_2 provide the same incorrect answer for a correspondence $c \in D$ is:

$$Pr(c \text{ is incorrect}|w_1 \perp w_2) = (1-r) \cdot (1-r) = (1-r)^2$$

As a result, the conditional probability of observing D is:

$$Pr(D|w_1 \perp w_2) = r^{2k_t}(1-r)^{2k_f} \quad (9)$$

Similarly, we consider the case where w_1 and w_2 belong to an imitating group. The probability that w_1 and w_2 provide the same correct answer for a correspondence $c \in D$ is:

$$Pr(c \text{ is correct}|w_1 \sim w_2) = r$$

And the probability that w_1 and w_2 provide the same incorrect answer for a correspondence $c \in D$ is:

$$Pr(c \text{ is incorrect}|w_1 \sim w_2) = (1-r)$$

Consequently, the conditional probability of observing D is:

$$Pr(D|w_1 \sim w_2) = r^{k_t}(1-r)^{k_f} \quad (10)$$

Put it altogether, we have a concrete calculation of eq. (8) as follows.

$$Pr(w_1 \sim w_2|D) = \left(1 + (\frac{1-\alpha}{\alpha}) r^{k_t}(1-r)^{k_f}\right)^{-1} \quad (11)$$

This equation captures several intuitions we expect in practice. For example, when the number of common false answers increases (k_f increases), the probability that two workers are dependent increases. This is because two independent workers rarely give all the same incorrect answers. Moreover, it should be noted that our method works better when the two given workers have more common questions.

Boundary Computation. Since the ground truth is unknown, we cannot compute the exact value of r, k_f, and k_t. However, this computation is unnecessary if we can bound the probability in eq. (11). For this purpose, we have two propositions as follows.

Proposition 1. $\forall \alpha \in [0,1], k_t \in \mathbb{N}, k_f \in \mathbb{N}$. We have:

$$Pr(w_1 \sim w_2|D) \geq \left(1 + \frac{1-\alpha}{\alpha} \frac{k_f^{k_f} k_t^{k_t}}{(k_f + k_t)^{k_f + k_t}}\right)^{-1} \quad (12)$$

Proof. The proof is given in the appendix. □

Proposition 2. $\forall \alpha \in [0,1]$, we have:

$$Pr(w_1 \sim w_2|D) \geq \left(1 + \frac{1-\alpha}{\alpha} \frac{|V|^{|V|}(|D|-|V|)^{|D|-|V|}}{(|D|)^{|D|}}\right)^{-1} \quad (13)$$

Proof. The proof is given in the appendix. □

Example 1. Consider two workers w_1 and w_2 who share three violations $V = \{v_1, v_2, v_3\}$, where each violation is joint and contains three correspondences ($|D| = 9$). A crowdsourcing system is vulnerable when there are enough imitating groups in the crowd. Thus, let us assume that $\alpha > 0.3$ (i.e. the probability that any two workers in the crowd belong to an imitating group is greater than 0.3). Then following the inequality in eq. (13) and having the function $\frac{1-x}{x}$ being monotonically decreasing with $x \in [0, 1]$, we have:

$$Pr(w_1 \sim w_2|D) \geq \left(1 + \frac{1 - 0.3}{0.3}\frac{3^3 6^6}{9^9}\right)^{-1} \approx 0.9925$$

In other words, we can conclude that these two workers belong to an imitating group with a probability greater than 0.99.

5. Answer Aggregation

As we explained, we demand to validate a given attribute correspondence from several crowd workers. In this section we explain how we aggregate the possibly different responses to a single value such that the (expected) quality of the aggregated value meets a predefined standard. In fact, we first estimate the error rate of any given aggregated value using our probabilistic models and then we show how to minimize the financial costs of obtaining a a set of correspondences of a given quality with the help of the integrity constraints.

5.1. Deriving Aggregated Value and Error Rate

We will derive the aggregated value of a correspondence based on the probability $Pr(X_c)$ that is the probability of a given correspondence is *true*. We compute these probabilities, as we explained in Section 3. We compute the aggregation decision $g_\pi(c)$ for each correspondence $c \in C$ that is a pair $g_\pi(c) = \langle a_c, e_c \rangle$, where a_c is the aggregated value (*true* or *false*) and e_c is the error rate. The aggregation decision is obtained as follows:

$$g_\pi(c) = \begin{cases} \langle true, 1 - Pr(X_c = true) \rangle, \text{ if } Pr(X_c = true) \geq 0.5 \\ \langle false, 1 - Pr(X_c = false) \rangle, \text{ otherwise} \end{cases} \quad (14)$$

The aggregated value in the aggregation decision thus corresponds to the value that has a higher probability (and lower error rate). The error rate is the probability of making wrong decision.

We would like to reduce this error rate, for each correspondence. We could achieve a lower error rate if we ask more questions, however asking more questions induces higher costs as well. Instead, we will try to lower the error rate given a limited budget of money with the help of the integrity constraints that we explain in the next section.

5.2. Leveraging Constraints to Reduce the Error Rate

For several crowdsourcing tasks, one could achieve a lower error rate through asking more questions [20, 21]. This is, in fact, a trade-off between the costs and the accuracy [22].

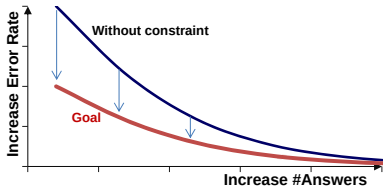

Figure 3. Optimization goal

Figure 3 depicts this situation: higher number of answers correspond to lower error rates. We would like to lower this error-rate curve as much as possible. If we can achieve the same error rate, with a lower number of answers then we can reduce the number of questions that is needed to achieve a given error rate. To achieve this goal, we leverage the network-level consistency constraints. In the following, we will show how to exploit these constraints and how can we aggregate the worker's responses in the presence of the constraints. These aggregation techniques require more complex probability estimations than those we presented in Section 3. While we present the precise definitions, we do not detail here how to compute these probabilities efficiently. We mention that there exists methods that enable to compute the required probabilities fast enough, such that our methods can be used in real application contexts.

Aggregating with Constraints. We extend here the definitions of Section 5.1, and include the effects of the integrity constraints to the calculation of aggregation decision. We will show that by using constraints, we need fewer answers to obtain an aggregated result with the same error rate. In other words, given the same set of answers on a certain correspondence, the error rate of aggregation with constraints is lower than those without constraints (i.e. Section 5.1).

Given the aggregation $g_\pi(c)$ of a correspondence c, we compute the justified aggregation $g^\gamma_\pi(c)$ when taking into account the integrity constraint γ. The aggregation $g^\gamma_\pi(c)$ is obtained similarly to equation 14, but we use here the conditional prbability $Pr(X_c|\gamma)$ instead of $Pr(X_c)$. Formally,

$$g^\gamma_\pi(c) = \begin{cases} \langle true, 1 - Pr(X_c = true|\gamma) \rangle, \text{ If } Pr(X_c = true|\gamma) \geq 0.5 \\ \langle false, 1 - Pr(X_c = false|\gamma) \rangle, \text{ Otherwise} \end{cases} \quad (15)$$

In the following, we describe how to obtain the conditional probabilities $Pr(X_c|\gamma)$ in the case of 1-1 constraint and the cycle constraint. Then, we show why

the use of constraints can reduce the error rate. We leave the investigation of other types of constraints as an interesting future work.

Aggregating with 1-1 Constraint. Our approach is based on the intuition illustrated in Figure 4(A), depicting two correspondences c_1 and c_2 with the same source attribute. After receiving the answer set from workers and applying the probabilistic model (section 5.1), we obtained the probability $Pr(X_{c_1} = true) = 0.8$ and $Pr(X_{c_2} = false) = 0.5$. When considering c_2 independently, it is hard to conclude c_2 being approved or disapproved. However, when taking into account c_1 and 1-1 constraint, c_2 tends to be disapproved since c_1 and c_2 cannot be *true* at the same time. Indeed, following probability theory, the conditional probability $Pr(X_{c_2} = false|\gamma_{1-1}) \approx 0.83 > Pr(X_{c_2} = false)$.

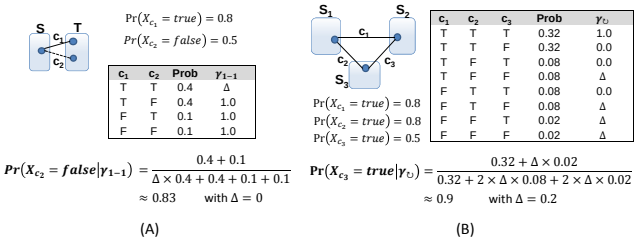

Figure 4. Compute conditional probability with (A) 1-1 constraint and (B) cycle constraint

Computing conditional probability. Given the same set of correspondences $\{c_0, c_1, \ldots, c_k\}$ above, let us denote p_i as $Pr(X_{c_i} = true)$ for short. Without loss of generality, we consider c_0 to be the favorite correspondence whose probability p_0 is obtained from the worker answers. Using the Bayesian theorem and equation 1, the conditional probability of correspondence c_0 with 1-1 constraint γ_{1-1} is computed as:

$$Pr(X_{c_0} = true|\gamma_{1-1}) = \frac{Pr(\gamma_{1-1}|X_{c_0} = true) \cdot Pr(X_{c_0} = true)}{Pr(\gamma_{1-1})} = \frac{(x + \Delta(1-x)) \times p_0}{y + \Delta(1-y)} \quad (16)$$

$$\text{where} \quad x = \prod_{i=1}^{k}(1-p_i) \qquad y = \prod_{i=0}^{k}(1-p_i) + \sum_{i=0}^{k}\left[p_i \prod_{j=0, j\neq i}^{k}(1-p_j)\right]$$

x can be interpreted as the probability of the case where all other correspondences except c being disapproved. y can be interpreted as the probability of the case where all correspondences being disapproved or only one of them being disapproved. The precise derivation of equation 16 is given in the appendix.

Theorem 1. The conditional probability of a correspondence c being false with 1-1 constraint is less than or equal to the probability of c being false without constraint. Formally, $Pr(X_c = false|\gamma_{1-1}) \geq Pr(X_c = false)$.

Proof. The proof can be found in the appendix. □

From this theorem, we can conclude that the error rate is reduced only when the aggregated value is *false*. From equation 14 and 15, the error rate with 1-1 constraint (i.e. $1 - Pr(X_c = false|\gamma_{1-1})$) is less than or equal to the one without constraint (i.e. $1 - Pr(X_c = false)$). In other words, the 1-1 constraint supports reducing the error rate when the aggregated value is *false*.

Aggregating with Cycle Constraint. To motivate our definitions we present a small matching network. Figure 4(B) depicts an example of cycle constraint for three correspondences c_1, c_2, c_3. After receiving the set of answers from workers and applying probabilistic model (section 5.1), we obtained the probability $Pr(X_{c_1} = true) = Pr(X_{c_2} = true) = 0.8$ and $Pr(X_{c_3} = true) = 0.5$. When considering c_3 independently, it is hard to conclude c_3 being *true* or *false*. However, when taking into account c_1, c_2 under the cycle constraint, c_3 tends to be *true* since the cycle created by c_1, c_2, c_3 shows an interoperability. Therefore, following probability theory, the conditional probability $Pr(X_{c_3} = true|\gamma_{1-1}) \approx 0.9 > Pr(X_{c_3} = true)$.

Computing conditional probability. Given a closed cycle along $c_0, c_1, \ldots, c_k$, let denote the constraint on this circle as $\gamma_{\circlearrowleft}$ and p_i as $Pr(X_{c_i} = true)$ for short. Without loss of generality, we consider c_0 to be the favorite correspondence whose probability p_0 is obtained by the answers of workers in the crowdsourcing process. Following the Bayesian theorem and equation 2, the conditional probability of correspondence c_0 with circle constraint is computed as:

$$Pr(X_{c_0} = true|\gamma_{\circlearrowleft}) = \frac{Pr(\gamma_{\circlearrowleft}|X_{c_0} = true) \times Pr(X_{c_0} = true)}{Pr(\gamma_{\circlearrowleft})} = \frac{(\prod_{i=1}^{k}(p_i) + \Delta(1-x)) \times p_o}{\prod_{i=0}^{k}(p_i) + \Delta(1-y)} \quad (17)$$

$$\textit{where} \quad x = \prod_{i=1}^{k}(p_i) + \sum_{i=1}^{k}\left[(1-p_i)\prod_{j=1, j\neq i}^{k} p_j\right] \qquad y = \prod_{i=0}^{k}(p_i) + \sum_{i=0}^{k}\left[(1-p_i)\prod_{j=0, j\neq i}^{k} p_j\right]$$

x can be interpreted as the probability of the case where only one correspondence among $c_1, \ldots, c_k$ except c_0 is disapproved. y can be interpreted as the probability of the case where only one correspondence among $c_0, c_1, \ldots, c_k$ is disapproved. The detail derivation of equation 17 is given in the appendix.

Theorem 2. Given a correspondence c together with other correspondences $c_1, \ldots, c_k$ creating a closed cycle

$\gamma_{\circlearrowleft} = \{c_0, c_1, \ldots, c_k\}$, the conditional probability $Pr(X_c = true|\gamma_{\circlearrowleft})$ is greater than or equal to the probability $Pr(X_c = true)$, $Pr(X_c = true|\gamma_{\circlearrowleft}) \geq Pr(X_c = true)$ if $\frac{1}{\Delta} \geq \sum_{i=1}^{k} \frac{1-p_i}{p_i}$.

Proof. The proof can be found in the appendix. □

Note that the condition of Δ is often satisfied since Δ closed to 0 and p_i closed to 1. From this theorem, we conclude that the error rate is reduced only when the aggregated value is *true*. With an appropriately chosen Δ, in equation 14 and 15, the error rate with cycle constraint (i.e. $1 - Pr(X_c = true|\gamma_{\circlearrowleft})$) is less than or equal to the one without constraint (i.e. $1 - Pr(X_c = true)$). In other words, circle constraint supports reducing the error rate when the aggregated value is *true*.

Aggregating with Multiple Constraints. In general settings, we could have a finite set of constraints $\Gamma = \{\gamma_1, \ldots, \gamma_n\}$. Let denote the aggregation with a constraint $\gamma_i \in \Gamma$ is $g_\pi^{\gamma_i}(c) = \langle a_c^i, e_c^i \rangle$, whereas the aggregation without any constraint is simply written as $g_\pi(c) = \langle a_c, e_c \rangle$. Since the constraints are different, not only could the aggregated value a_c^i be different ($a_c^i \neq a_c^j$) but also the error rate e_c^i could be different ($e_c^i \neq e_c^j$). In order to reach a single decision, the challenge then becomes how to define the multiple-constraint aggregation $g_{\pi_\Gamma}^{\Gamma}(c)$ as a combination of single-constraint aggregations $g_\pi^{\gamma_i}(c)$.

Since the role of constraints is to support reducing the error rate and the aggregation $g_\pi(c)$ is the base decision, we compute the multiple-constraint aggregation as $g_\pi^\Gamma(c) = \langle a_c, e_c^\Gamma \rangle$, where $e^\Gamma = min(\{e_c^i | a_c^i = a_c\} \cup e_c)$. We take the minimum of error rates in order to emphasize the importance of integrity constraints, which is the focus of this work. Therefore, the error rate of the final aggregated value is reduced by harnessing constraints. For the experiments with real datasets described in the next section, we will show that this aggregation reduces half of worker efforts while preserving the quality of aggregated results.

6. Experiments

The main goal of the following comprehensive experimental evaluation is to demonstrate the usage of crowdsourcing and integrity constraints in reconciling schema matching results in a matching network. To verify the effectiveness of our proposed methods, four experiments are performed: (i) evaluations on detecting spammers, (ii) relationship between the error rate and the matching accuracy, (iii) effects of spammer ratio on termination of crowdsourcing process, and (iv) evaluations on answer aggregation. We proceed to report the results on the real datasets using both real workers and simulated workers. The results highlight that the presented approach supports reconciling schema matching by effectively using crowdsourcing harnessing integrity constraints.

6.1. Experimental Settings

Datasets. We have used many real-world datasets spanning various application domains, from Web forms to business schemas observed in data marketplaces.

- *Business Partner (BP):* The set comprises database schemas that model business partners in enterprise systems.
- *PurchaseOrder (PO):* We extracted purchase order e-business schemas from various resources.
- *University Application Form (UAF):* We extracted schemas from Web interfaces of American university application forms.
- *WebForm:* The schemas for this dataset have been automatically extracted from Web forms using OntoBuilder [23].

These datasets are publicly available [24] and descriptive statistics for the schemas are given in Table 1. In the experiments, the topology of schema matching network is a complete graph (i.e., all graph nodes are interconnected with all other nodes). To generate candidate correspondences, we used two well-known schema matchers (with default parameters), COMA++ [25, 26] and AMC [27]. All experiments ran on an Intel Core i7 system (2.8GHz, 4GB RAM).

Table 1. Datasets

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

Table 2. Constraint violations

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

Integrity Constraints. For demonstration purposes, we consider the two integrity constraints, the one-to-one constraint and the cycle constraint, cf., Section 2.1. Table 2 lists the number of constraint violations among the candidate correspondences generated by the matchers. Rather independent of the applied schema matcher, we observe a large number of violations. Hence, there is a clear need for an efficient and effective crowdsourcing framework. All constraint violations are detected before-hand in the experiments.

Crowd Simulation. Since workers have wide-ranging levels of expertise, using real crowdsourcing services cannot cover all scenarios. We also develop a simulation engine that generates simulated workers to show the effectiveness of our approach in different settings.

In our simulation, we assume that the ground truth is known in advance (i.e. the ground truth is known for the experimenter, but not for the (simulated) crowd worker). Many previous studies [18, 28] characterized different types of crowd workers to reflect their expertise. Based on the classification in [18], we simulate 5 worker types as depicted in Figure 5. (1) *Experts:* who have deep knowledge about specific domains and answer questions with very high reliability. (2) *Normal workers:* who have general knowledge to give correct answers, but with few occasional mistakes. (3) *Sloppy workers:* who have very little knowledge and thus often give wrong answers, but unintentionally. (4) *Uniform spammers:* who intentionally give the same answer for all their own questions. (5) *Random spammers:* who carelessly give the random answer for any question. To model these types of workers, we use two parameters: *sensitivity*—the proportion of actual positives that are correctly identified—and *specificity*—the proportion of negatives that are correctly identified. Following the statistical result in [28], we set randomly the sensitivity and specificity of each type of workers as follows. For experts, the range is $[0.9, 1]$. For normal workers, it falls into $[0.6, 0.9]$. For sloppy workers, the range $[0.1, 0.4]$ is selected. For random spammers, it varies from 0.4 to 0.6. Especially for uniform spammers, there are two regions: (i) $sensitivity \in [0.8, 1], specificity \in [0, 0.2]$ and (ii) $sensitivity \in [0, 0.2], specificity \in [0.8, 1]$.

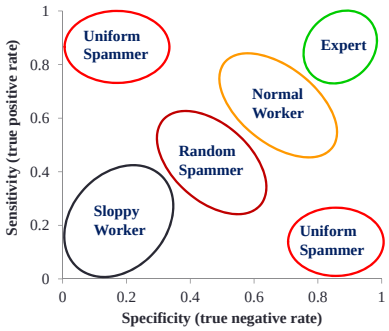

Figure 5. Characterization of worker types

Evaluation metrics. We rely on the following evaluation measures.

- *Matching Precision & Matching Recall:* To measure the quality of matching results, we rely on an exact matching G (which contains correct correspondences validated before-hand). Formally, the precision and recall of a set of correspondences V are $MPrec(V)=(|V \cap G|)/|V|$ and $MRec(V)=(|V \cap G|)/|G|$, where G is the exact matching (i.e. ground truth) given by the dataset provider.
- *Detecting Precision & Detecting Recall:* To measure the quality of spammer detection techniques, we define precision and recall of their detection results. Given W_S as the set of spammers in the crowd and W_D as the set of detected spammers, we have: $DPrec=(|W_D \cap W_S|)/|W_D|$ and $DRec=(|W_D \cap W_S|)/|W_S|$.

6.2. Evaluations on detecting spammers

In this set of experiments, we would like to evaluate the effectiveness of our worker assessment methods, including spammer detection and imitating group detection.

Detecting Spammers. In this experiment, we study our proposed method of detecting spammers as described in Section 4.1. To do so, we create a population of 100 simulated workers, among which 20% are experts, 35% are normal workers, 45% are sloppy workers. Such a worker expertise distribution has been observed at real-world crowdsourcing services [18]. In our experiments we vary gradually the ratio of spammers.

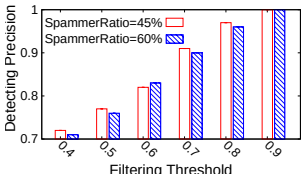

Figure 6. Effects of Spammer Ratio on Precision

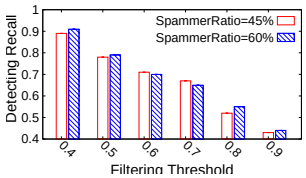

Figure 7. Effects of Spammer Ratio on Recall

Figure 6 and 7 depict the results, that we have obtained as an average over 100 runs of our simulation. The X-axis is the filtering threshold α (that is the threshold value above which we consider a worker a spammer), varying from 0.4 to 0.9. The Y-axes is the detecting precision and detecting recall. We experimented with a spammer ratio of 45% and of 60% (while keeping the distribution of non-spammers, i.e. expert, normal and sloppy workers unchanged).

An interesting finding is that there is a tradeoff between detecting precision and detecting recall. When we raise the filtering threshold, the detecting precision increases while the detecting recall decreases. This is because other workers might also give the answers that create violations. Therefore, on the one hand, if the filtering threshold is too low, the detection mechanism could return both spammers and non-spammers, leading to low precision but high recall. On the other hand, if the filtering threshold is too high, the detection could not return all spammers, leading to high precision but low recall.

Another noticeable observation is that the detecting precision and recall are not affected by the ratio of spammers in the crowd. More precisely, the average

difference of precision and recall between the two settings (SpammerRatio=60% and SpammerRatio=45%) are 0.01 and 0.02, respectively. This is reasonable since our detection mechanism calculates the violation rate for each worker independently.

Detecting Imitating Groups. In this experiment, we study our proposed method for detecting imitating groups as described in Section 4.2. To this end, we create a population of 100 simulated workers, in which there are 20% experts, 35% are normal workers, 45% are sloppy workers. To simulate the imitating groups, we modify this worker population as follows. Denote r_g is a predefined imitating ratio in the crowd. A copier (who imitates answers from other workers) is simulated by randomly choosing one of $1 - r_g$ independent workers and copying one or many his answers. Two workers are called belonging to an imitating group if one copies from another (one independent and one copier) or both of them copy from a same worker (two copiers).

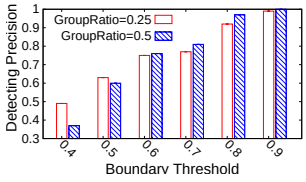

Figure 8. Effects of Imitating Group Ratio on Precision

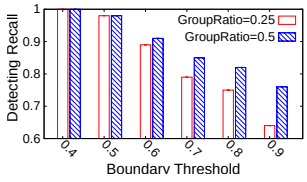

Figure 9. Effects of Imitating Group Ratio on Recall

Figure 8 and 9 illustrate the results. There are 100 correspondences given to workers for validation. The X-axis is boundary threshold, varying from 0.4 to 0.9. For each pair of workers, if the imitating probability between them (eq. (13)) is greater than the boundary threshold, they are detected as belonging to an imitating group. The Y-axes is the detecting precision and detecting recall based on the detected workers. In our simulation experiments we used imitating ratio r_g of 25% and of 50%.

An interesting finding is that there is also a trade-off between detecting precision and detecting recall. Similar to the previous experiment, increasing the boundary threshold would increase the precision but lowering the recall. This is because a worker who does not imitate might still give some answers similar to the others (by accidence or the answers are the common ground themselves).

Another noticeable observation is that the detecting precision and recall are sensitive to the imitating ratio. More precisely, the difference between the two cases (r_g = 25% and r_g = 50%) are considerable (0.3 for precision and 0.5 for recall). This can be explained by the fact that an imitating group might contain many workers. As the size of imitating groups is larger, the belonging worker is more likely to be detected by one or many other workers in the same group (the detection considers every pair of workers). In other words, more true-positive cases are detected as the group ratio is larger, leading to higher precision and recall.

6.3. Relationship between Error Rate and Matching Accuracy

In order to assess the matching accuracy, we borrow the *precision* metric from information retrieval, which is the ratio of ($true$) correspondences existing in ground truth among all correspondences whose aggregated value is $true$. However, the ground truth is not known in general. Therefore, we use an indirect metric—error rate—to estimate the matching quality. We expect that the lower error rate, the higher quality of matching results.

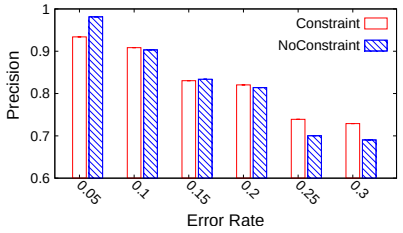

Figure 10. Relationship between error rate and precision

The following empirical results aim to validate this hypothesis. We conduct the experiment on all real datasets with the population of 100 simulated workers as above (20% experts, 35% normal workers, 45% spammers). Since the purpose of this experiment is to study the relationship between error rate and matching accuracy only, we do not consider spammers and imitating groups in the crowd.

Figure 10 depicts the resulting relationship of the error rate and precision, which is averaged over all datasets. In that, we vary error threshold ϵ from 0.05 to 0.3, meaning that the questions are posted to workers until the error rate of aggregated value is less than the given threshold ϵ. The precision is plotted as a function of ϵ. We aggregate the worker answers by two strategies: without constraint and with constraint. Here we consider both 1-1 constraint and cycle constraint as hard constraints, thus $\Delta = 0$.

The key observation is that when the error rate is decreased, the precision approaches to 1. Reversely, when the error rate is increased, the precision is reduced but greater than $1 - \epsilon$. Another interesting finding is that when the error rate is decreased, the value distribution of precision in case of with and without constraint is quite similar. This indicates that our method of updating the error rate is relevant.

In summary, the error rate is a good indicator of the quality of aggregated results. Since the ground truth

is hidden, our goal was to verify if the error rate is a useful metric for matching quality. The result indicated that there was no significant difference between the two metrics. In terms of precision, the quality value is always around $1 - \epsilon$. In other words, the error threshold ϵ can be used to control the real matching quality.

6.4. Evaluations on Answer Aggregation

In Section 5.2, we already saw the benefit of using constraints in reducing error rate. In other words, with given requirement of low error, the constraints help to reduce the *expected cost* of crowd validation (i.e. the number of questions that need to be asked the workers). In this set of experiments, we will study the effects of harnessing such constraints on real datasets (BP, PO, WebForm, etc.).

We will analyze the effectiveness of constraints in three different settings: (i) effects of worker population, (ii) effects of spammers, and (iii) effects of imitating groups. All the settings have a common process as follows. Given an error threshold ($\epsilon = 0.1$), we iteratively post questions to workers and aggregate the worker answers (with and without constraints) until the error rate is less than ϵ. After the process ends, we instantiate an aggregated matching that consists of correspondences aggregated as correct (i.e. exists in ground truth). The reported numbers are averaged over all datasets.

Effects of Worker Reliability. We create a population of 100 simulated workers with 55% normal workers and 45% spammers. We use the detection method in Section 4.1 to detect the spammers and remove their answers from the answer set (we choose the filtering threshold = 0.6 as it balances the accuracy trade-off as in Section 6.2). The results are presented in Figure 11. The Y-axes is the expected cost, the matching precision and the matching recall of the aggregated matching. The X-axis is the reliability (denoted as r) of normal workers in the population, varying from 0.6 to 0.8. The reliability of a worker is $r = \frac{2sensitivity \times specificity}{sensitivity + specificity}$, where *sensitivity* and *specificity* of a worker are already described in the experimental setting.

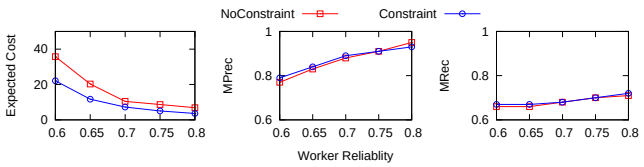

Figure 11. Effects of worker reliability on answer aggregation

A significant observation in the results is that for all values of error threshold and worker reliability, the expected cost of the aggregation with constraints is definitely smaller (approximately a half) than the case without constraints. For example, with worker reliability is $r = 0.6$, the expected number of questions is reduced from 35 (without constraints) to 22 (with constraints). This concludes the fact that the constraints help to reduce the error rate, and subsequently reduce the expected cost.

Another key finding in Figure 11 is that both the matching precision and matching recall improve when the worker reliability increases. This implies that our detection method works well and the aggregated matching does not affected by spammers. Moreover, it is worth noting that the differences in MPrec and MRec between using constraints and not using constraints are not significant, since both cases are computed for the same error threshold ($\epsilon = 0.1$).

Effects of Spammers. We use the same worker population like the above experiment, except that we fix the reliability of normal workers to 0.75. Similar to the spammer detection experiment in Section 6.2, we increase the spammer ratio from 45% to 60% to study its effect on the expected cost and matching quality. The results are presented in Figure 12. The X-axis is the ratio of spammers in the crowd. The Y-axes are the expected cost, matching precision, and matching recall.

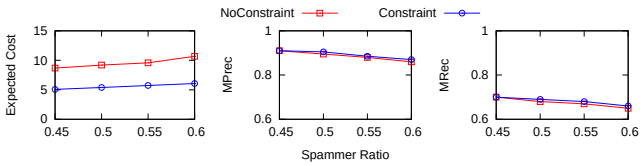

Figure 12. Effects of spammers on answer aggregation

An interesting finding is that when the spammer ratio increases, the expected cost only slightly increases. Moreover, the *Constraint* case always has a lower cost than the *NoConstraint* case. This supports the fact that our detection method is effective and insensitive to the spammer ratio. Although there are many spammers (who often give incorrect answers which increase the error rate) in crowd, most of them are detected and prevented from increasing the expected cost.

Another highlighted observation is that both the matching precision and matching recall only slightly decrease when the spammer ratio increases. This is reasonable and can be explained the same as above: our detection method is not significantly affected by the number of spammers.

Effects of Imitating Groups. We use the same worker population like the above experiment, in which we also fix the reliability of normal workers to 0.75. Similar to the imitating group detection experiment in Section 6.2, we increase the imitating ratio to study its effects on the expected cost and matching quality. The results are

showed in Figure 13. The X-axis is the ratio of workers belonging to imitating groups in the crowd, varying from 30% to 60%. The Y-axes are the expected cost, matching precision, and matching recall.

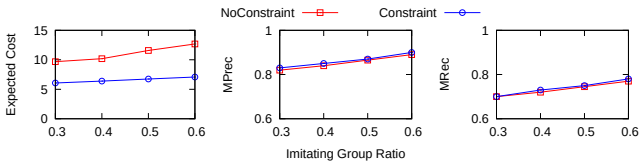

Figure 13. Effects of imitating groups on answer aggregation

A key finding is that when the number of imitating groups increases, the expected cost slightly decreases. This can be explained by the experimental result in Section 6.2: the detection is more effective when the size of imitating groups increases. This leads to early filtering of incorrect answers; and thus, reduce the error rate (or equivalently reduce the expected cost). It is worth noting that the *Constraint* case still has a lower cost than the *NoConstraint* case. This concludes the fact that the both cases are not affected by imitating groups since most of these groups are always detected.

Another interesting observation is that the matching precision and matching recall slightly increase when the imitating ratio increases. This is straightforward to understand. The more malicious workers are likely to be removed, the higher chances that we get a better aggregated matching.

Moreover, it is worth nothing that the matching precision and matching recall in all the three experimental settings have a similar trend since we do not add new correspondences in the validation process.

7. Related Work

We now review salient work in schema matching and crowdsourcing areas that are related to our research.

7.1. Schema matching

Database schema matching is an active research field. The developments of this area have been summarized in two surveys [6, 29]. Existing works on schema matching focused mainly on improving quality parameters of matchers, such as precision or recall of the generated matchings. Recently, however, ones started to realize that the extent to what 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 [30], systematic matching ensemble selection [31] or automatic tuning of the matcher parameters [32].

While there is a large body of works on schema matching, the post-matching reconciliation process (that is central to our work) has received little attention in the literature. Recently, there are some works [33–36] using pay-as-you-go integration method that establishes the initial matching and then incrementally improves matching quality. While the systems in [33, 34] rely on one user only, the framework in [35, 37] relies on multiple users.

7.2. Schema matching network

The idea of exploiting the presence of a large set of schemas to improve the matchings has been studied before. Holistic matching [38] attempted to exploit statistical co-occurrences of attributes in different schemas and use them to derive complex correspondence. Whereas, corpus-based matching [39] attempted to use a 'corpus' of schemas to augment the evidences that improve existing matchings and exploit constraints between attributes by applying statistical techniques. Network level constraints, in particular the circle constraints, were originally considered in [12, 40] in which they studied the establishment of semantic interoperability in a large-scale P2P network. There are several applications based on schema matching networks in particular and schema matching in general, including schema reuse [41], web search [42], and Deep Web [43].

In this paper, we study contextual information and integrity constraints (e.g., 1-1 and circle constraints) on top of the schema matching network. A somewhat related concept of alignment space was introduced in Euzenat [44]. The authors consider a network of ontologies and alignments between them, which is similar to our setting, however unlike us, they do not consider network-aware consistency conditions. The alignment space is mainly used for designing similarity-measures.

7.3. Crowdsourcing

In recent years, crowdsourcing has become a promising methodology to overcome human-intensive computational tasks. Its benefits vary from unlimited labor resources of user community to cost-effective business models. The book [13] summarized problems and challenges in crowdsourcing as well as promising research directions for the future. A wide range of crowdsourcing platforms, which allows users to work together in a large-scale online community, have been developed such as Amazon Mechanical Turk and CloudCrowd.

On top of these platforms, there are also many crowdsourcing applications that have been built for specific domains. For example, in [22], the crowdsourcing is employed to validate the search results of automated image search on mobile devices. In [45], the authors leveraged the user CAPTCHAs inputs in web forms to recognize difficult words that cannot

solved precisely by optical character recognition (OCR) programs.

Similar to our work, the authors of [46] also make use of crowdsourcing to validate the correspondences and reduce their uncertainty. However, they only focus on a pair-wise matching and using entropy-based decision strategy to maximize the uncertainty reduction at a single validation step. Whereas, our work leverages integrity constraints on top of a schema matching network to reduce the overall validation effort.

Regarding the utilization of constraints, there are some previous works such as [47, 48]. In [47], the constraints were used to define the tasks for collaborative planning systems whereas in [48], the constraints were used to check worker quality by quantifying the consistency of worker answers. In our work, the constraints are used to provide evidences for detecting malicious workers and adjust the error rate for reducing worker efforts.

Deutch et al. [49] also use integrity constraints, in particular database key constraints to identify problems with the data that is collected through crowdsourcing. They also propose repair mechanisms for the cases where the user input violates the constraints.

8. Conclusions and Future Work

We have presented our techniques to reconcile a schema matching network through crowdsourcing. The algorithms take the correspondences generated by automatic schema matchers as input and they generate validation questions. The crowd workers respond to these questions by indicating whether a particular attribute correspondence should be accepted or rejects. We can express various natural expectations about the network in the form of integrity constraints. We demonstrated that these constraints can be exploited in various ways: they enable to improve the worker assessment methods, they can be used to reduce the necessary validation efforts. Moreover they are also usful for spam detection. We demonstrated these desirable properties through our experiments on real datasets, with the help of a simulated crowd population.

Our work opens up several future research directions. First, one can extend our notion of schema matching network and consider representing other integrity constraints (e.g., functional dependencies or domain-specific constraints). Second, one can devise more applications which could be transformed into the schema matching network. While our work focuses on schema matching, our techniques, especially the constraint-based aggregation method, can be applied to other tasks such as entity resolution, business process matching, or Web service discovery.

Acknowledgement. This research has received funding from the NisB project - European Union's Seventh Framework Programme (grant agreement number 256955) and the PlanetData project - Network of Excellence (grant agreement number 257641). The work of Zoltán Miklós was partially supported by the project Aresos, financed by CNRS through the mastodons programme. The authors thank for anonymous referees for their helpful suggestions.

References

[1] Gonzalez, H., Halevy, A.Y., Jensen, C.S., Langen, A., Madhavan, J., Shapley, R., Shen, W. *et al.* (2010) Google fusion tables: web-centered data management and collaboration. In *SIGMOD*: 1061–1066.

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

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

[4] Das Sarma, A., Fang, L., Gupta, N., Halevy, A., Lee, H., Wu, F., Xin, R. *et al.* (2012) Finding related tables. In *SIGMOD*: 817–828.

[5] Bernstein, P., Madhavan, J. and Rahm, E. (2011) Generic Schema Matching, Ten Years Later. In *VLDB*.

[6] Rahm, E. and Bernstein, P.A. (2001) A Survey of Approaches to Automatic Schema Matching. *JVLDB* : 334–350.

[7] Nguyen, Q.V.H., Wijaya, T.K., Miklos, Z., Aberer, K., Levy, E., Shafran, V., Gal, A. *et al.* (2013) Minimizing Human Effort in Reconciling Match Networks. In *ER*.

[8] Nguyen, Q.V.H., Nguyen, T.T., Miklós, Z. and Aberer, K. (2013) On leveraging crowdsourcing techniques for schema matching networks. In *DASFAA*: 139–154.

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

[10] Gal, A. (2011) *Uncertain Schema Matching* (Morgan & Claypool).

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

[12] Cudré-Mauroux, P., Aberer, K. and Feher, A. (2006) Probabilistic message passing in peer data management systems. In *ICDE*: 41.

[13] von Ahn, L. (2009) Human computation. In *Design Automation Conference*: 418 –419.

[14] Quoc Viet Hung, N., Tam, N., Tran, L. and Aberer, K. (2013) An evaluation of aggregation techniques in crowdsourcing. In *WISE*: 1–15.

[15] Dawid, A.P. and Skene, A.M. (1979) Maximum likelihood estimation of observer error-rates using the EM algorithm. *J. R. Stat. Soc.* : 20–28.

[16] Gao, N., Webber, W. and Oard, D. (2014) Reducing reliance on relevance judgments for system comparison by using expectation-maximization. In *Advances in Information Retrieval*, 1–12.

[17] Hosseini, M., Cox, I.J., Milić-Frayling, N., Kazai, G. and Vinay, V. (2012) On aggregating labels from multiple

crowd workers to infer relevance of documents. In *Advances in information retrieval*, 182–194.
[18] Vuurens, J., de Vries, A. and Eickhoff, C. (2011) How much spam can you take? an analysis of crowdsourcing results to increase accuracy. In *CIR*.
[19] Difallah, D.E., Demartini, G. and Cudre-Mauroux, P. (2012) Mechanical cheat: Spamming schemes and adversarial techniques on crowdsourcing platforms. In *CrowdSearch*.
[20] Ipeirotis, P.G., Provost, F. and Wang, J. (2010) Quality management on amazon mechanical turk. In *HCOMP*: 64–67.
[21] Parameswaran, A.G., Garcia-Molina, H., Park, H., Polyzotis, N., Ramesh, A. and Widom, J. (2012) Crowdscreen: algorithms for filtering data with humans. In *SIGMOD*: 361–372.
[22] Yan, T. and Kumar, V. (2010) CrowdSearch: exploiting crowds for accurate real-time image search on mobile phones. *8th international conference on Mobile* : 77–90.
[23] Roitman, H. and Gal, A. (2006) Ontobuilder: fully automatic extraction and consolidation of ontologies from web sources using sequence semantics. In *ICSNW*: 573–576.
[24] http://lsirwww.epfl.ch/schema_matching.
[25] Do, H.H. and Rahm, E. (2002) COMA - A System for Flexible Combination of Schema Matching Approaches. In *VLDB*: 610–621.
[26] Aumueller, D., Do, H.H., Massmann, S. and Rahm, E. (2005) Schema and ontology matching with coma++. In *SIGMOD*: 906–908.
[27] Peukert, E., Eberius, J. and Rahm, E. (2011) AMC - A framework for modelling and comparing matching systems as matching processes. In *ICDE*: 1304–1307.
[28] Kazai, G., Kamps, J. and Milic-Frayling, N. (2011) Worker types and personality traits in crowdsourcing relevance labels. In *CIKM*: 1941–1944.
[29] Bernstein, P.A., Madhavan, J. and Rahm, E. (2011) Generic Schema Matching, Ten Years Later. *PVLDB* : 695–701.
[30] Duchateau, F., Coletta, R., Bellahsene, Z. and Miller, R.J. (2009) (not) yet another matcher. In *CIKM*: 1537–1540.
[31] Gal, A. and Sagi, T. (2010) Tuning the ensemble selection process of schema matchers. *JIS* : 845–859.
[32] Lee, Y., Sayyadian, M., Doan, A. and Rosenthal, A.S. (2007) eTuner: tuning schema matching software using synthetic scenarios. *JVLDB* : 97–122.
[33] Jeffery, S.R., Franklin, M.J. and Halevy, A.Y. (2008) Pay-as-you-go user feedback for dataspace systems. In *SIGMOD*: 847–860.
[34] Qi, Y., Candan, K.S. and Sapino, M.L. (2007) Ficsr: feedback-based inconsistency resolution and query processing on misaligned data sources. In *SIGMOD*: 151–162.
[35] McCann, R., Shen, W. and Doan, A. (2008) Matching Schemas in Online Communities: A Web 2.0 Approach. In *ICDE*: 110–119.
[36] Nguyen, Q.V.H., Nguyen, T.T., Miklós, Z., Aberer, K., Gal, A. and Weidlich, M. (2014) Pay-as-you-go reconciliation in schema matching networks. In *ICDE*: 220–231.
[37] Nguyen, H.Q.V., Luong, X.H., Miklós, Z., Quan, T.T. and Aberer, K. (2013) Collaborative schema matching reconciliation. In *CoopIS*: 222–240.
[38] Su, W., Wang, J. and Lochovsky, F. (2006) Holistic schema matching for web query interfaces. In *EDBT*: 77–94.
[39] Madhavan, J., Bernstein, P.A., Doan, A. and Halevy, A. (2005) Corpus-based schema matching. In *ICDE*: 57–68.
[40] Aberer, K., Cudré-Mauroux, P. and Hauswirth, M. (2003) Start making sense: The Chatty Web approach for global semantic agreements. *JWS* : 89–114.
[41] Hung, N.Q.V., Tam, N.T., Aberer, K. *et al.* (2014) Privacy-preserving schema reuse. In *DASFAA*: 234–250.
[42] He, H., Meng, W., Yu, C. and Wu, Z. (2004) Automatic integration of web search interfaces with wise-integrator. *JVLDB* : 256–273.
[43] Wu, W., Yu, C., Doan, A. and Meng, W. (2004) An interactive clustering-based approach to integrating source query interfaces on the deep web. In *SIGMOD*: 95–106.
[44] David, J., Euzenat, J. and Sváb-Zamazal, O. (2010) Ontology similarity in the alignment space. In *International Semantic Web Conference (1)*: 129–144.
[45] von Ahn, L., Maurer, B., McMillen, C., Abraham, D. and Blum, M. (2008) recaptcha: Human-based character recognition via web security measures. *Science* : 1465–1468.
[46] Zhang, C.J., Chen, L., Jagadish, H.V. and Cao, C.C. (2013) Reducing uncertainty of schema matching via crowdsourcing. In *PVLDB*: 757–768.
[47] Zhang, H., Law, E., Miller, R., Gajos, K., Parkes, D. and Horvitz, E. (2012) Human computation tasks with global constraints. In *CHI*: 217–226.
[48] Chen, K.T., Wu, C.C., Chang, Y.C. and Lei, C.L. (2009) A crowdsourceable qoe evaluation framework for multimedia content. In *MM*: 491–500.
[49] Deutch, D., Greenshpan, O., Kostenko, B. and Milo, T. (2012) Declarative platform for data sourcing games. In *Proceedings of the 21st International Conference on World Wide Web*, WWW '12: 779–788.

Appendix

Sketch proof for proposition 1: Based on Cauchy's inequality, we have:

$$(\frac{k_f}{k_t})^{k_t} r^{k_t} (1-r)^{k_f} \leq \left(\frac{k_t \frac{k_f^r}{k_t} + k_f(1-r)}{k_f + k_t} \right)^{k_f+k_t} \leq \frac{k_t^{k_f+k_t}}{(k_f+k_t)^{k_f+k_t}}$$

Hence,

$$r^{k_t}(1-r)^{k_f} \leq \frac{k_f^{k_f} k_t^{k_t}}{(k_f+k_t)^{k_f+k_t}}$$

Using this inequality with eq. (11) completes the proof.

Sketch proof for proposition 2: Let n and m be two positive integers with $0 < m < n$. Given $x \in [m, n-m]$, it can be easily seen that $f(x) = x^x(n-x)^{n-x}$ is a convex function. Thus, w we

have:

$$x^x(n-x)^{n-x} \leq max(f(m), f(n-m)) = m^m(n-m)^{n-m}$$

We will apply this inequality with $m = |V|$, $n = |D|$, and $x = k_f$. By definition, we have $k_t + k_f = |D|$. Moreover, we have the fact that each violation has at least one correct correspondence and one incorrect correspondence; i.e. $m \leq k_f, k_t \leq n$. As a result, we have:

$$\frac{k_f^{k_f} k_t^{k_t}}{(k_f + k_t)^{k_f+k_t}} = \frac{k_f^{k_f}(|D| - k_f)^{|D|-k_f}}{(|D|)^{|D|}} \leq \frac{|V|^{|V|}(|D| - |V|)^{|D|-|V|}}{|D|^{|D|}}$$

Using this inequality with eq. (12) completes the proof.

Computing the conditional probability $Pr(X_{c_0}|\gamma_{1-1})$**:** According to Bayes theorem, $Pr(X_{c_0}|\gamma_{1-1}) = \frac{Pr(\gamma_{1-1}|X_{c_0}) \times Pr(X_{c_0})}{Pr(\gamma_{1-1})}$. Now we need to compute $Pr(\gamma_{1-1})$ and $Pr(\gamma_{1-1}|X_{c_0})$. Let denote $p_i = Pr(X_{c_i} = true)$, for short. In order to compute $Pr(\gamma_{1-1})$, we do following steps: (1) express $Pr(\gamma_{1-1})$ as the sum from the full joint of $\gamma_{1-1}, c_0, c_1, \ldots, c_k$, (2) express the joint as a product of conditionals. Formally, we have:

$$\begin{aligned} Pr(\gamma_{1-1}) &= \textstyle\sum_{c_0,c_1,\ldots,c_k} Pr(\gamma_{1-1}, X_{c_0}, X_{c_1}, \ldots, X_{c_k}) \\ &= \textstyle\sum Pr(\gamma_{1-1}|X_{c_0}, X_{c_1}, \ldots, X_{c_k}) \times Pr(X_{c_0}, X_{c_1}, \ldots, Xc_k) \\ &= 1 \times Pr(X_{c_0}, X_{c_1}, \ldots, X_{c_k}|m(X_{c_0}, X_{c_1}, \ldots, X_{c_k}) \leq 1) \\ &+ \Delta \times Pr(X_{c_0}, X_{c_1}, \ldots, X_{c_k}|m(X_{c_0}, X_{c_1}, \ldots, X_{c_k}) > 1) \\ &= y + \Delta \times (1 - y) \end{aligned}$$

where $m()$ counts the number of X_{c_i} assigned as *true*
$y = \prod_{i=0}^{n}(1-p_i) + \sum_{i=0}^{n}[p_i \prod_{j=0,j\neq i}^{n}(1-p_j)]$

Similar to computing $Pr(\gamma_{1-1})$, we also express $Pr(\gamma_{1-1}|X_{c_0})$ as the sum from the full joint of $\gamma_{1-1}, c_1, \ldots, c_k$ and then express the joint as a product of conditionals. After these steps, we have $Pr(\gamma_{1-1}|X_{c_0} = true) = x + \Delta \times (1 - x)$, where $x = \prod_{i=1}^{k}(1 - p_i)$. After having $Pr(\gamma_{1-1})$ and $Pr(\gamma_{1-1}|X_{c_0})$, we can compute $Pr(X_{c_0}|\gamma_{1-1})$ as in equation 16.

Computing the conditional probability $Pr(X_{c_0}|\gamma_{\circlearrowleft})$**:** According to Bayes theorem, $Pr(X_{c_0}|\gamma_{\circlearrowleft}) = \frac{Pr(\gamma_{\circlearrowleft}|X_{c_0}) \times Pr(X_{c_0})}{Pr(\gamma_{\circlearrowleft})}$. In order to compute $Pr(\gamma_{\circlearrowleft}|X_{c_0})$ and $Pr(\gamma_{\circlearrowleft})$, we also express $Pr(\gamma_{\circlearrowleft}|X_{c_0})$ as the sum from the full joint of $\gamma_{1-1}, c_0, c_1, \ldots, c_k$ and then express the joint as a product of conditionals. After some transformations, we can obtain equation 17.

Sketch proof for theorem 1: From equation 16, we can obtain $y = x + \sum_{i=1}^{k}[p_i \prod_{j=0,j\neq i}^{k}(1-p_j)]$. Since $\sum_{i=1}^{k}[p_i \prod_{j=0,j\neq i}^{k}(1-p_j)] \geq 0$ and $\Delta \leq 1$, we have $x + \Delta(1-x) \leq y + \Delta(1-y)$. Following this inequality and equation 16, we conclude $Pr(X_c = true|\gamma_{1-1}) \leq Pr(X_c = true)$.

Sketch proof for theorem 2: After some transformations, we can derive that $Pr(X_c = true|\gamma_{\circlearrowleft}) \geq Pr(X_c = true)$ is equivalent to $(1-p_0)\prod_1^k p_i \geq \Delta(x-y)$. Moreover, we have $x - y = (1-p_0)\sum_{i=1}^{k}[(1-p_i)\prod_{j=1,j\neq i}^{k} p_j]$. Therefore, we conclude $Pr(X_c = true|\gamma_{\circlearrowleft}) \geq Pr(X_c = false)$ if $\frac{1}{\Delta} \geq \sum_{i=1}^{k}\frac{1-p_i}{p_i}$.