A Crowdsourcing Approach for Finding Misidentifications of Bibliographic Records

Atsuyuki Morishima[1], Tomita Shiori[1], Takanori Kawashima[2], Takashi Harada[3], Norihiko Uda[1], Sho Sato[3] and Yukihiko Abematsu[1]

[1] University of Tsukuba

[2] National Diet Library

[3] Doshisha Univeristy

Abstract

Because there is no perfect technique for automatic identification of bibliographic records, cleaning the identification results manually is indispensable. However, to recruit human resources for the task is often difficult. This paper discusses a microtask-based crowdsourcing approach to the problem. An important issue is to design a good strategy for generating tasks to be assigned to workers, maintaining the quality and reducing the number of tasks. In this study, we explore a design space defined by two criteria to reduce the number of assigned microtasks for finding misidentifications caused by automatic identification techniques. We compare four task-generation strategies using bibliographic records of the National Diet Library. One of the strategies reduced 55.7% of tasks from the baseline strategy and statistic analysis showed that the quality of its result is comparable to those of the other three strategies.

Keywords: crowdsourcing, bibliographic records, human-powered join

Citation: Morishima, A., Shiori, T., Kawashima, T., Harada, T., Uda, N., Sato, S., & Abematsu, Y. (2014). A Crowdsourcing Approach for Finding Misidentifications of Bibliographic Records. In iConference 2014 Proceedings (p. 177–191). doi:10.9776/14061

Acknowledgements: The authors are grateful to the contributors of Crowd4U, whose names are partially listed at http://crowd4u.org. This research was partially supported by PRESTO from the Japan Science and Technology Agency, and by the Grant-in-Aid for Scientific Research (#25240012) from MEXT, Japan.

Contact: mori@slis.tsukuba.ac.jp

1 Introduction

Recently, there have been many attempts to construct *union catalogs* by collecting and joining bibliographic records originally managed by different organizations. However, it is often the case that one source (e.g., a book) is represented by more than one bibliographic record. Therefore, it is common practice to conduct automatic identification of bibliographic records in which algorithms identify the records that represent the same source. For example, the National Diet Library (NDL) in Japan conducts automatic identification for constructing its union catalog.

A common approach to automatic identification is to use algorithms for computing the *identification key* for each record, and we determine that records having the same key represent the same source. In many cases, algorithms of automatic identifications use ISBNs to compute identification keys, because they are the only numbers that exist in virtually all bibliographic records.

However, it is well known that there is no perfect technique for automatic identification of bibliographic records. There are several reasons for this. First, the contents of different records representing the same source are not necessarily the same if their creators are different; attributes such as title may have spelling variants. It may miss some words or may have values that should be entered for other attributes. For example, Figure 1 shows two apparently different records that represent the same source. The strings in the title attributes are slightly different from each other, and the string in the series attribute of the second record is an abbreviation. NDL also has many record pairs in which both represent the same source but one is written in Japanese while the other is in English. Note that the machine can determine that two records represent the same source *only if* both the identification keys and other information in the records

are exactly the same. If not, the machine can conclude that two sources (books) are neither different nor the same, because it is difficult for the machine to determine whether different words and sentences in two records are semantically equivalent to each other.

Second, the values used to compute identification keys are often inappropriately assigned to the sources. For example, a recent research by NDL found that it is not rare for different sources to have the same ISBN (Publishing a detailed report on the topic is one of our future work). A reason is that publishers often *reuse* the same ISBN for different publications. Another reason is that bibliographic records tend not to be updated after an incorrect ISBN is given to the record. Keys derived from such inappropriate numbers lead to *misidentifications* by algorithms of automatic identification, i.e., different sources are incorrectly determined to be the same source.

Therefore, it is necessary to manually verify the automatically identified records for final identification. Humans are good at determining whether the difference in two records implies a misidentification, because they can tell whether two different expressions are semantically equivalent. However, the number of records that require manual verification can be large, and it is often difficult to recruit human resources for the task.

Title	Series	Publisher
Towards the e-society : e-commerce, e-business, and e-government : the first IFIP Conference on E-Commerce, E-Business, E-Government (13E 2001), October 3-5, 2001, Zurich, Switzerland / edited by Beat Schmid, Katarina Stanoevska-Slabeva, Volker Tschammer	The International Federation for Information Processing ; 74	Kluwer Academic Publishers
Towards the e-society: e-commerce, e-business, and e-government : the first IFIP conference on e-commerce, e-business, e-government (13E 2001) October 3-5, 2001, Zurich, Switzerland. : Oct 2001, Zurich, Switzerland	IFIP ; 74	Kluwer Academic Publishers

Figure 1: Bibliograhic records having the same ISBN

This paper discusses a crowdsourcing approach to the problem that is taken by L-Crowd project (http://crowd4u.org/lcrowd). To our knowledge, L-Crowd is one of the largest crowdsourcing projects for library-related problems; the project involves core members and collaborators from more than 16 universities and the NDL of Japan.

In the project, we crowdsource performing *microtasks* designed for solving library-related problems. Here microtasks are tasks that can be performed in a short period of time. For example, Figure 2 is a microtask that asks a human whether the two faces in the photographs are the same person.

Microtask-based crowdsourcing is a popular form of crowdsourcing, with many microtask-based crowdsourcing platforms such as Amazon's Mechanical Turk. In general, a microtask-based crowdsourcing platform has a *task pool* into which *requesters* register microtasks that will be assigned to *workers*. L-Crowd uses a microtask-based crowdsourcing platform named Crowd4U (http://crowd4u.org) (Morishima, Shinagawa, Mitsuishi, Aoki, & Fukusumi, 2012). Crowd4U is deployed at universities, and many anonymous or registered volunteers perform the microtasks registered in Crowd4U's task pool.

The contributions of the paper are as follows:

(1) Crowdsourcing for identification of bibliographic records. To our knowledge, this paper is the first to discuss microtask-based crowdsourcing in the identification of bibliographic records. There are attempts to use crowdsourcing to solve library-related problems. For example, in the "Civil War Faces" project, the Library of Congress crowdsources the tagging of photos through Flickr

(http://www.flickr.com/photos/library_of_congress/sets/72157625520211184/). Australian National Library crowdsources proofreading the results of applying OCR to old newspapers (http://trove.nla.gov.au/ndp/del/home). Bodleian Library of University of Oxford crowdsources enhancing metadata of music scores (http://whatsthescoreatthebodleian.wordpress.com/). Our approach is unique in the following two ways. First, we use human power to find semantic equivalence between different expressions, while others use it to recognize patterns in images. Second, we discuss the strategies for generating tasks in a scientific way, while we have not noticed any other projects that gave their scientific justifications for their design decisions. This paper is the first to present a formal framework for the application of crowdsourcing to the problem of finding misidentifications.

(2) Novel technique for generating microtasks. When we apply microtask-based crowdsourcing, what constitutes an appropriate design of microtasks and how to generate them is an important issue. The design and generation strategy of microtasks affects both the number of necessary microtasks to reach the goal and the quality of the output data. This paper explores a design space of microtask-based crowdsourcing that is defined by two criteria for finding misidentifications of bibliographic records. One of the two criteria, called the *contraction*, is a novel technique we proposed (Tomita, Morishima, Uda, & Harada, 2013) for extending the design space. In general, *contraction* is a technique in graph theory to merge different nodes into one in a given graph (Wilson, 2010), and is often used to reduce the size of the graph without losing some important properties of the graph. We show that the technique is effective in reducing the number of microtasks in the problem of finding misidentifications, keeping data quality acceptable.

(3) Experiments with NDL data. We conducted an experiment using a real set of NDL bibliographic records to compare four variations of microtask generation strategies within the design space. As explained, the NDL conducted automatic identification for constructing its union catalog and suffered from the problem of misidentifications. Our experimental results suggest that crowdsourcing is promising in a real setting. In addition, we plan to use the proposed scheme to obtain data to improve the query results of bibliographic search in other libraries in Japan.

The remainder of this paper is as follows. Section 2 explains related work. Section 3 formalizes our problem as a human-powered join. Section 4 presents four strategies for generating microtasks in a design space defined by two independent criteria. Section 5 shows the results of our experiments to compare the four task-generation strategies, and Section 6 is the summary.

2 Related Work

Our problem is related to various topics from different domains. This section enumerates some of them.

Automatic Identification of Bibliographic Records. Automatic identification techniques compute *identification keys* from bibliographic records and conclude that two records represent the same source if they have the same keys. For example, a study attempted to develop appropriate identification keys for *Unicanet*, a well-known union catalog network in Japan (http://unicanet.ndl.go.jp/psrch/redirect.jsp?type=psrch). Another example is the NDL search (http://iss.ndl.go.jp/), which implements automatic identification techniques using several variations of identification key. Both work utilize ISBN as part of the identification keys. However, publishers do not necessarily assign ISBNs appropriately. We plan to apply our technique to find misidentifications due to inappropriate assignment of ISBNs and analyze the obtained results for improving the quality of identification keys.

Efficient Processing of Human-powered Operations. Strategies for generating microtasks are an important factor for successful crowdsourcing. For example, in some cases, workers tend to avoid performing a complex task even if they are paid more than the sum of the price for each component of the original task (Kittur, Smus, Khamkar, & Kraut, 2011). In essence, our problem can be modeled as a human-powered join (Marcus, Wu, Karger, Madden, & Miller, 2011), an operation to determine whether a pair of data items satisfy a given condition. In our setting, the condition is that two bibliographic records with the same ISBN represent different sources. Efficient processing of human-powered joins is discussed in existing papers (Marcus, Wu, Karger, Madden, & Miller, 2011; Mitsuishi, Morishima, Shinagawa, & Aoki, 2013). Proposed techniques include (1) changing the size of tasks; (2) changing the presentation of microtasks; and (3) using the power of crowd to reduce the number of tasks.

Recently, Wang et al. independently proposed to consider transitive relations to reduce the number of required tasks in human-powered joins (Wang, Li, Kraska, Franklin, & Feng, 2013). The proposed technique is related to our contraction technique in the sense that they use the results for some pairs to infer the results of other pairs. One of the contributions of our paper is that it shows our design space for task-generation strategies that incorporates this family of optimization techniques is effective in finding misidentifications of bibliographic records, a real problem in the library domain.

Data Quality in Crowdsourcing. Another important issue in the data-centric crowdsourcing is data quality. Since many techniques for improving data quality are independent of the task design, our proposed technique can be combined with many existing techniques. For example, many crowdsourcing adopt majority voting (Marcus, Wu, Karger, Madden, & Miller, 2011), a technique that relies on the law-of-large-numbers. In Section 5, we show that one of our task-generation strategies significantly reduces the number of tasks and the quality of outputs is comparable to the others when we adopt majority voting. Another approach is a coordination game (Morishima, Shinagawa, & Mochizuki, 2011), in which rational workers give appropriate values. Jain and Parkes (2008) provides a game-theoretic analysis of games with a purpose for obtaining data, and shows that a simple change of the incentive structure can affect the obtained data.

3 Identification of Bibliographic Records

In this section, we first briefly explain the problem of identification of bibliographic records. Then, we formalize our problem as a human-powered join.

3.1 Identification of Bibliographic Records

Given two bibliographic records b_i and b_j, identification of bibliographic records is to determine whether b_i and b_j represent the same source.

This can be done manually by experts, or automatically by machines (algorithms) if bibliographic records are machine-readable. The latter is called *automatic identification of bibliographic records*. A general approach for automatic identification is to compute *identification keys* using data encoded in bibliographic records and then to determine that two bibliographic records refer to the same source if they have the same key. Typically, identification keys are computed using ISBNs (ISO 2108:2005) and MARC numbers (ISO 2709:2008), which are assigned by publishers and MARC management institutes, respectively, for distinguishing publications. However, at present, ISBNs are the only numbers that exist in virtually all bibliographic records. Other numbers, such as MARC numbers, are not in common use compared to ISBNs. Therefore, they are used as a supplementary means in automatic identification techniques.

The problem is that the results of automatic identification techniques are not necessarily correct, because it is impossible to construct a perfect identification key (Taniguchi, 2009). There are several reasons for this. First, because the bibliographic records are created manually, the created records often have minor variations or are just incorrect. Second, even if the information written in the records is correct, ISBNs,

which in many cases are used to construct identification keys, are often inappropriately assigned by publishers to sources so that they cannot work as perfect identifiers.

3.2 Formalization

Given that the results of automatic identification of bibliographic records are not necessarily correct, and that we need to use human power to solve the problem, we apply a crowdsourcing approach to the problem of finding misidentifications. In this section, we formalize our problem as a human-powered join (Marcus, Wu, Karger, Madden, & Miller, 2011). A human-powered join is a special form of the join operation of database relations (tables). Logically, the join operation first enumerates every pair of tuples (rows) of two relations and then selects from the candidate pairs those pairs that satisfy the given condition, which will be included in the results. In a human-powered join, humans determine whether each pair of data items satisfies the given condition. For each pair of data items, a task is invoked to ask a worker whether a pair of data items satisfies a condition. For example, assume that we have a relation of photos of human faces and want to self-join the relation to find pairs that has photos of the same person. Then, Figure 2 is an example of a microtask that asks a worker if the persons in two photos are the same.

Note that the model is compatible with the formalization of the record linkage problem presented in (Gu, Baxter, Vickers, & Rainsford, 2003), in which the problem is modeled as computing the Cartesian product of sets of records and then checking whether two records in each pair match with each other. Modeling our problem as a human-powered join generalizes this formalization so that the process of checking whether the condition holds is partly crowdsourced.

Bibliographic Records. We use a relation with a relational schema B(tid,record) to store a set of bibliographic records (Figure 3). Here, tid is a tuple identifier and record is a relational attribute to store each bibliographic record. Note that tid is not an identifier of sources represented by records, but that of *tuples* in the relation.

Automatic Identification. Each technique for automatic identification of bibliographic records using identification keys can be modeled as a self-join of Relation B as follows:

$$B \bowtie_{key(record)=key(record') \wedge tid<tid'} B'$$

Here B′ is an alias of Relation B to distinguish two relations in the self-join. key(v) is a function defined by each automatic identification technique to compute identification keys. As mentioned, each identification key is computed using values contained in record in automatic identification techniques. We need tid < tid′ to avoid re-evaluating the same record pair, i.e., to avoid evaluating the pair (tid_2,tid_1) if the pair (tid_1,tid_2) has already been evaluated.

Figure 2: Example of a task for a human-powered join (photos are taken from JAFFE Database (Lyons, Akamatsu, Kamachi, & Gyoba, 1998))

tid	record
1	(Towards the e-society:..., ..., X publisher)
2	(The XML book, ... , ..., Y company)
3	(Towards e-society:, ..., ..., X publishing)

Figure 3: Example of a relation B(tid, record)

Finding misidentification by human-powered joins.

Then, the process of finding misidentification can be written as a human-powered join as follows.

$$B \bowtie_{key(record)=key(record')\wedge tid<tid'\wedge diff(record,record')} B'$$

Here diff(record,record′) is the condition the workers examine to see if the pair of records satisfies. The condition holds if record and record′ represent different sources (books). Therefore, the result of the human-powered join will contain the set of record pairs, each of which has two records with the same key, but considered by workers to represent different sources.

Definition of the same source. Assume that b_1 and b_2 have the same identification key. We define that b_1 and b_2 represent the same source if they are equivalent at the *expression level* defined by FRBR (IFLA Study Group on the Functional Requirements for Bibliographic Records, 1998). In other words, diff(b_1,b_2) holds if b_1 and b_2 represent sources that are different to each other at the expression level. The reason we chose the expression-level equivalence is that ISBNs and MARC numbers are assumed to be unique at this level.

4 Strategies for Generating Microtasks

In our approach, we need to generate microtasks for asking workers whether diff(record,record′) holds for given record and record′. In this section, we introduce two criteria to define the space for designing strategies, and explain four strategies within it.

To discuss the strategies, we introduce *groups* of bibliographic records. A group G_k is a set of bibliographic records with the identification key k and is defined as follows:

$$G_k = \{record | B(tid, record), key(record) = k\}.$$

For a given G_k, each task strategy generates a set of microtasks to obtain the set $\{(r,r') | r,r' \in G_k, \text{diff}(r,r')\}$. Therefore, we can discuss the generation of microtasks for a particular given group without losing generality. In the following discussions, we assume that k is an integer and that we have $G_1 = \{r_1,r_2,r_3,r_4\}$

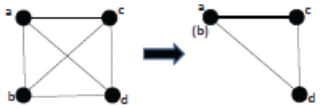

Figure 4: Contraction

4.1 Task Template and the Design Space

Our design space is defined by a task template for microtasks and two criteria to affect the number of generated tasks. We first explain the task template and then explain two criteria, namely, simultaneous

comparison and contraction. A task generation strategy is good if it requires a smaller number of tasks but the quality of the results remain comparable to, or better than, the others.

Task template. We use BTask(r,R) to denote a *task template* to generate microtasks for finding misidentifications. Here r is a bibliographic record and R is a set of records to be compared to r. Tasks are instantiated by the task template with parameters. For example, let r_1,r_2, and r_3 be bibliographic records. Then, BTask(r_1,{r_2,r_3,r_4}) is a task for asking workers whether any of r_2, r_3, and r_4 are different from r_1. Figure 8 is a screenshot of BTask(r_1,{r_2,r_3,r_4}).

First criterion: size of the task. The first criterion is the size of R. For pairwise comparison, we set $|R| = 1$. For example, we use BTask(r_1,{r_2}) to discover whether diff(r_1,r_2) holds. On the other hand, tasks for simultaneous comparison can be implemented by setting $|R| > 1$. For example, from the result for BTask(r_1,{r_2,r_3}), we can obtain information on both diff(r_1,r_2) and diff(r_1,r_3).

Second criterion: contraction. The second criterion is whether we apply *contraction* to the bibliographic records in the process of the human-powered join. The *contraction* is a technique in graph theory to merge different nodes into one in a given graph, and the edges connected to the deleted node will be inherited by the merged node. In our context, we use the technique to merge two bibliographic records into one if we know the two records represent the same source. This is done to reduce the number of comparisons without changing the result of the human-powered join. For example, assume that we have a group that has four bibliographic records and requires six comparisons (Figure 4). If we could merge record b to record a, the number of comparisons would be reduced to three, and the comparison results for the removed edges would be derived from the results of comparing a to c and d.

We developed four strategies for generating microtasks in the design space. First, the simplest strategy, A1, uses pairwise comparison and does not adopt contraction. Second, B1 uses the simultaneous comparison and does not adopt contraction. Third, A2 performs pairwise comparisons with the contraction technique. Finally, B2 performs simultaneous comparisons with the contraction technique.

4.2 A1: The Simplest Strategy

A1 is the simplest strategy that uses the pairwise comparison and does not adopt contraction. Figure 6 shows an example of an A1 task. The generation process is as follows. First, for each G_k, A1 constructs a set P_k of pairs such that $P_k = B \bowtie_{key(record)=k \wedge tid<tid'} B'$. For G_1, $P_1 = \{(r_1,r_2),(r_1,r_3),(r_1,r_4),(r_2,r_3),(r_2,r_4),(r_3,r_4)\}$. In general, $|p_k| = |_{|Gk|}C_2|$. Note that we can use P_k as an intermediate result to compute our human-powered join defined in Section 3.2, i.e., the join result can be computed by $\sigma_{diff(ri,rj)}(P_k)$. In other words, computing P_k is part of the process of our human-powered join.

Then, the second step is to generate microtasks to ask workers which pairs in P_k satisfy diff(r_i,r_j). A1 generates BTask(r_i,{r_j}) for each pair (r_i,r_j) in P_k. In our example, we obtain six A1 tasks because $|P_k| = 6$.

Figure 5 shows the algorithm used to generate A1 tasks, where a task is generated (Line 4) for each pair in P_k (Line 3).

```
1. Set of BTasks generateTasks(P_k) {
2.  tasks = {};
3.      for each (r_i, r_j) in P_k {
4.          tasks = tasks + {BTask(r_i, {r_j})};
5.      }
6.  return tasks;
7. }
```

Figure 5: Algorithm to generate A1 tasks

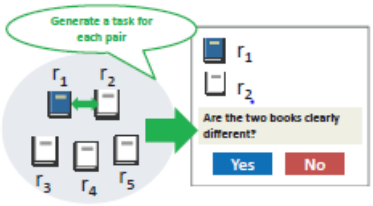

Figure 6: Generating an A1 task

4.3 B1: Simultaneous Comparison

B1 allows the simultaneous comparison but does not adopt contraction. B1 uses tasks represented by BTask(r,R) where R ≥ 1. For example, Figure 8 is BTask(r_1,{r_2,r_3,r_4}). B1 takes a pre-fixed parameter m that represents the maximum size of R. If we generate B1 tasks with m = 3 for the set G_1 of bibliographic records, we get three B1 tasks BTask(r_1,{r_2,r_3,r_4}), BTask(r_2,{r_3,r_4}), and BTask(r_3,{r_4}). Note that we can obtain from the results of the three tasks whether diff(r_i,r_j) holds for every pair in P_1.

Figure 7 is the algorithm to generate B1 tasks given P_k and m. In short, we enumerate every pair (r_i,R_i) such that (1) r_i is a record that appears on the left side of a pair in P_k (Line 3) and (2) R_i is a set of m records each r_j of which appears as (r_i,r_j) in P_k (Lines 4-7). For each such a pair, we generate BTask(r_i,R) (Line 7).

To obtain the m records, the algorithm utilizes a stack to store r_j that is paired with p_i (Lines 5, 7). In Lines 10-11, it generates a task when we have n ($0 < n < m$) records remaining in the stack for r_i. The algorithm does not generate tasks to produce duplicate results because P_k contains no (r_j,r_i) if it contains (r_i,r_j).

```
 1. Set of BTasks generateTasks(P_k, m) {
 2.   tasks = {};
 3.   for each r_i in P_k.leftRecords {
 4.     for each (r_i, r_j) in P_k {
 5.       stack.push(r_j);
 6.       if(stack.length == m){
 7.           tasks = tasks + {BTask(r_i, {stack.allpop})};
 8.       }
 9.     }
10.     if(stack.length > 0)
11.         tasks = tasks + {BTask(r_i, {stack.allpop})};
12.   }
13.  return tasks;
14. }
```

Figure 7: Algorithm to generate B1/B2 tasks

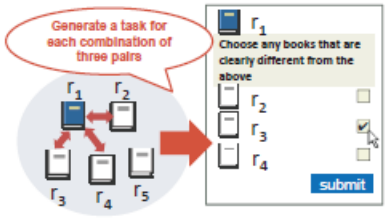

Figure 8: Example of a B1 task

4.4 A2 and B2: Introducing Contraction

A2 (B2) is the same as A1 (B1) except that it adopts the contraction technique for reducing the number of tasks. If a worker determines that diff(r_i,r_j) does not hold for the given pair (r_i,r_j), we consider r_i and r_j are equivalent in the sense that the results of tasks involving r_i are the same as those involving r_j. Therefore, we can omit the tasks involving r_j and reduce the number of tasks.

The A2 tasks are generated in the following way. First, we construct the set P_k of pairs in the same way as for A1 tasks. As with A1, the remaining step is to generate microtasks to discover whether each pair in P_k satisfies diff(r_i,r_j). During the process, we use the result of already performed tasks to remove pair (r_k,r_l) in P_k if we know whether diff(r_k,r_l) holds from other results. Finally, we stop the process if there is no pair (r_i,r_j) in P_k for which we do not know whether diff(r_i,r_j) holds.

Figure 9 shows the algorithm to generate tasks in A2. It generates a task for each pair in P_k (Lines 3-4), but if the result of a performed task suggests that the two records represent the same source, (1) the algorithm computes another pair p that would produce duplicate results (Line 9), (2) keeps p and the original pair in set `equiv` (Line 10), and (3) removes p from P_k (Line 11). Finally, `equiv` is used to produce the results for the removed pairs (Lines 16-18).

We can obtain the algorithm for B2 by slightly changing the algorithm in Figure 9. First, we call the algorithm shown in Figure 7 to produce the initial set of microtasks. Second, we insert a call for the same algorithm between Lines 11 and 12 to re-generate the set of microtasks. Finally, in Line 4, we register the generated tasks (instead of BTask(r_i,{r_j})) to the task pool. We omit the detail because it is straightforward.

```
 1. result={}  // stores the result set of pairs.
 2. equiv={}  // a set of pairs whose results are the same.
 3. for each (r_i, r_j) in P_k {
 4.   TaskPool.register(BTask(r_i, {r_j}).
 5.   Let isdiff be the result of BTask(r_i, {r_j}).
 6.   if(isDiff) result.add((r_i, r_j));
 7.   else { // two records represent the same source
 8.     for each r_q s.t. r_q.tid > r_i.tid
 9.       if(p in P_k s.t. p=(r_q, r_i) or (r_i, r_q)) {
10.          equiv.add((p, (r_q, r_j)));
11.          p_k.remove(p);
12.       }
13.     }
14.   }
15. }
16. for each (p, p') in equiv {
17.    if(p' in result) result.add(p);
18. }
```

Figure 9: Algorithm of A2

5 Experiment

This section explains the results of our experiment. The purpose of the experiment is twofold. First, we want to know whether the microtask-based crowdsourcing approach is applicable to the problem of finding misidentification of bibliographic records in a real setting. Second, we want to understand how changing task-generation strategies affects the process and results. In the experiment, we compared four task-generation strategies by the number of tasks, the elapsed times for performing tasks, and the quality of data in terms of precision and recall. Overall, we concluded that crowdsourcing is applicable to our problem, and B2 significantly reduced the number of tasks while keeping the quality of its result comparable to that of the other strategies.

5.1 Settings

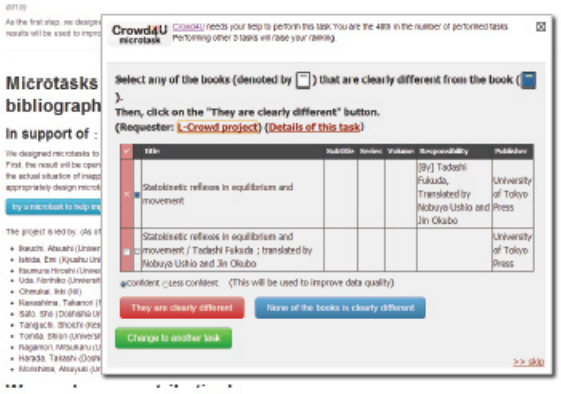

Figure 10: Screenshot of a task on Crowd4U

Crowdsourcing Platform. We used Crowd4U (http://crowd4u.org), a microtask-based crowdsourcing platform for academic purposes. Crowd4U is deployed at many universities in Japan, with anonymous and registered workers performing microtasks registered in its task pool. Figure 10 shows a screenshot of a Crowd4U microtask.

Data. In the experiment, we used bibliographic records of the unified catalog of the NDL. Prior to our experiment, the NDL applied automatic identification using ISBNs as identification keys to their records. Then, they selected the records whose ISBNs are not unique.

Given the set of records, we selected bibliographic records for a source written in Japanese. In the experiment, the native language of all workers was Japanese and we did not want to introduce another factor to compare different microtask designs. As a result, we obtained 34,254 records with 11,509 different ISBN groups. Table 1 shows the distribution of sizes of the groups.

Then, we conducted random sampling to extract 3% from the original groups, considering the distribution of the sizes of the ISBN groups. As a result, we obtain 341 groups with 933 records. We computed P_k for each $G_k(1 \leq k \leq 341)$ and obtained $\Sigma_{1 \leq k \leq 341}|P_k| = 1{,}315$.

$\|G_k\|$	Number of groups	Percentage (%)
2	8,479	73.67
3	1,129	9.81
4	546	4.74
5	388	3.37
6	260	2.26
7	162	1.41
8	136	1.18
9	83	0.72
10	103	0.89
11	47	0.41
12	39	0.34
13	24	0.21
14	13	0.11
≥ 15	100	0.8
Total	11,509	100

Table 1: Distribution of sizes of groups

5.2 Method

We first generated the tasks for A1 and B1 for the original data set to compare the number of generated tasks. Since our purpose was not to find the best parameters, we set m to three in this experiment. Finding the best parameters is an important future study. Then, we constructed sets of tasks for A1 and B1 for the groups with sizes more than two, because both A1 and B1 generate the same set of tasks if the size of the group is two. We carefully examined the records and found that the set contained several inappropriate records, which had set-ISBNs. Set-ISBNs are ISBNs assigned not to books but to the *set* of books, and not intended to be used to identify individual books. We removed 27 pairs with set-ISBNs from $\cup P_k$. Then, we manually created the answer set $result_{ans}$ of the human-powered join, i.e., the set of record pairs representing misidentifications. As a result, we got $\Sigma_{1 \leq k \leq 341}|P_k| = 1{,}034$, $|result_{ans}| = 737$ and $\Sigma_{1 \leq k \leq 341}|P_k| - |result_{ans}| = 297$. In Section 5.3, we use the former set of pairs to evaluate the elapsed time, while we use the latter set to evaluate the quality of results. Then, we inserted the generated tasks into the Crowd4U task pool for crowdsourcing. Finally, we used the results of A1 and B1 tasks to compute the results of tasks for A2 and B2, by removing the tasks to be removed by A2 and B2 algorithms. Note that this is possible because each task is independently performed on the crowdsourcing platform, and removing some tasks does not affect the results of others.

5.3 Results and Discussions

This section compares the four variations in terms of the number of generated tasks, elapsed time for performing tasks, and quality of results.

Number of tasks. Figure 11 compares the number of generated tasks. As expected, both simultaneous comparison and contraction are effective in reducing the number of tasks. Compared with A1 tasks, A2, B1, and B2 tasks were reduced by 27.5%, 43.7%, and 55.7%, respectively.

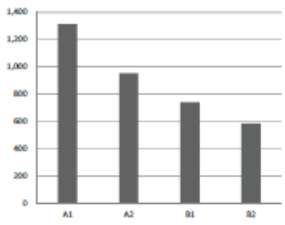

Figure 11: Number of generated tasks

Elapsed time. Figure 12 shows the elapsed times required for performing the tasks. The figure shows the averages, medians, and modes of elapsed times for performing A1 and B1 tasks (Times for A2 and B2 are omitted because they are the same as those for A1 and B1). Here we did not include the results of two tasks that we failed to log the elapsed times for. Note that in Figure 12, there are large differences between the average times and the medians. This suggests that there are outliers. A most likely reason is that workers often performed other jobs while performing a task. Therefore, we applied the technique proposed by Tukey (1977) to remove outliers. The technique uses the box-and-whisker plot and the interquartile range (IQR) as a parameter, which denotes the difference between the first and third quartiles. We used 1.5IQR to detect outliers. A1 and B1 (excluding outliers) in Figure 12 show the times after removal.

Overall, the figure suggests that the elapsed time for A1 tasks is shorter than that for B1 tasks. This is because B1 tasks require workers to perform simultaneous comparisons that are more difficult than pairwise comparisons. However, the elapsed times for both types of tasks are significantly below 10s, and short enough for microtask-based crowdsourcing.

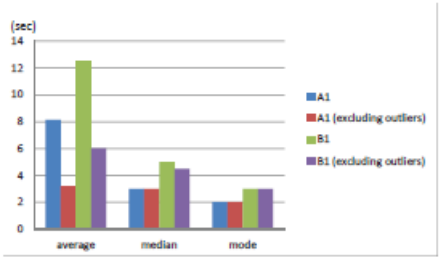

Figure 12: Elapsed times for performing tasks

Quality of the results. Finally, we compared the quality of the results. Table 2 shows the results of human-powered joins with the four task-generation strategies. Here result_s is a set of record pairs determined by workers as misidentifications with the task-generation strategy s. Then, we computed recall, precision, and F-measures using result_{ans}.

Figure 13 shows recalls and precisions obtained by the human-powered joins with the four task-generation strategies. The results show that all recalls and precisions are more than 0.85 and 0.9, respectively. Table 3 shows F-measures, all of which are above 0.9. Figure 14 compares the number of required tasks and F-measures, where each point corresponds to a task-generation strategy.

Strategy s	$\|result_s\|$	$\Sigma_k\|P_k\|$-$\|result_s\|$
A1	674	360
A2	692	342
B1	660	374
B2	680	354
Answer Set	737	297

Table 2: Number of tuples in the result

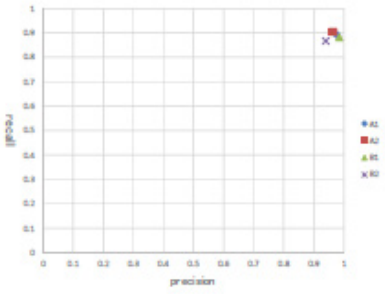

Figure 13: Recalls and precisions

	A1	B1	A2	B2
F-measure	0.933	0.934	0.931	0.902

Table 3: F-measures

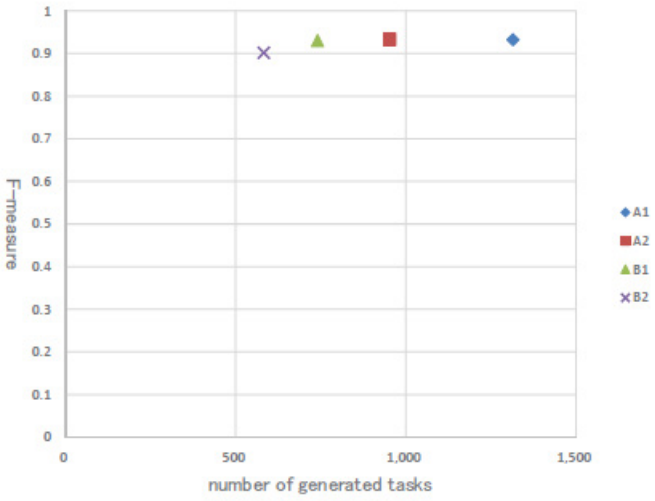

Figure 14: F measures and #tasks

The result shows that if we directly used the result, the quality of the result of B2 tasks would be lower than those of others. More precisely, tests for differences among the proportions show that the differences between recalls of A2 and B2 are significant at the 5% significance level. For precisions, the differences for pairs (B1, A2), (A2, B2), (B1, B2) and (A1, B2) are significant at the 5% significance level. Note that the result does not suggest a negative correlation between the number of tasks and data quality. In our experiment, the result of B2 (A2) tasks are derived from those of B1 (A1) tasks. When we adopt the contraction technique, the quality of the result is heavily affected by the result of microtasks performed in the earlier stage. Therefore, the results imply that the quality of the result of the earlier B1 microtasks was lower than that of A1 tasks. We can conclude that it is important to guarantee the quality of task results in the early stage of the process.

In reality, however, it is rare to directly adopt the result of performing a task by one worker. It is rather typical to integrate results by majority voting. Then, the differences in qualities become much smaller. For example, if we assume that the probability is uniform and each task is performed by three workers, we can expect that F-measures of A1, A2, B1, and B2 are between 0.9996 and 0.9991. B2 outperforms the others in the sense that it significantly reduces the number of tasks and its output is comparable to that of the other strategies. Our statistical analysis showed no significant difference in precision and recall if we use majority voting by five and seven workers, respectively.

6 Summary

This paper applied microtask-based crowdsourcing for finding misidentification in the results of automatic identification of bibliographic records. We modeled the problem as a human-powered join and considered four variations of task-generation strategies in a design space defined by two criteria. The first is the number of records to be compared at once, and the second is whether we apply contraction, a novel technique to optimize human-powered joins. We compared four task-generation strategies using bibliographic records of the NDL. The experimental result showed that one of the strategies reduced 55.7% of tasks from the baseline strategy and statistic analysis showed that the quality of its result was comparable to that of the other three strategies.

Future studies include the detailed analysis of the design space with different parameter settings. It also includes the development of a method to incorporate techniques to improve data quality, and the development of a method to combine the power of experts and crowdsourcing, where disputed results are passed to experts for detailed verification.

7 References

Gu, L., Baxter, R., Vickers, D., & Rainsford, C.(2003). Record linkage: Current practice and future directions. *CMIS Technical Report 03/83, CSIRO Mathematicaland Information Sciences.*

IFLA Study Group on the Functional Requirements for Bibliographic Records.(1998). *Functional requirements for bibliographic records: Final report.* Munchen: K.G. Saur.

Jain, S., & Parkes, D. C.(2008). A Game-Theoretic Analysis of Games with a Purpose. *WINE 2008*, 342-350.

Kittur, A., Smus, B., Khamkar, S., & Kraut, R. E.(2011). CrowdForge: Crowdsourcing Complex Work. *UIST'11*, 43-52.

Lyons, M. J., Akamatsu, S., Kamachi, M., & Gyoba, J.(1998). Coding Facial Expressions with Gabor Wavelets. *Third IEEE International Conference on Automatic Face and Gesture Recognition*, 200-205.

Marcus, A., Wu, E., Karger, D. R., Madden, S., & Miller, R.C.(2011). Human-powered Sorts and Joins. *PVLDB 5*(1), 13-24.

Mitsuishi, T., Morishima, A., Shinagawa, N., & Aoki, H.(2013). Efficient Evaluation of Human-powered Joins with Crowdsourced Join Prefilters. *ACM ICUIMC2013*, 6 pages.

Morishima, A., Shinagawa, N., Mitsuishi, T., Aoki, H., & Fukusumi, S.(2012). CyLog/Crowd4U: A Declarative Platform for Complex Data-centric Crowdsourcing. *PVLDB 5*(12), 1918-1921.

Morishima, A., Shinagawa, N., & Mochizuki, S.(2011). The Power of Integrated Abstraction for Data-Centric Human/Machine Computations. *VLDS 2011*, 7-10.

Taniguchi, S.(2009). Automatic Identification of "Works" toward Construction of FRBRized OPACs: an Experiment on JAPAN/MARC Bibliographic Records. *Library & information science 61*(2009), 119-151. *(in Japanese).*

Tomita, S., Morishima, A., Uda, N., & Harada, T.(2013). Design of Croudsourcing Tasks for Finding Bibliographic Misidentifications. *IEICE Annual Conference.(in Japanese).*

Tukey, J. W.(1977). Exploratory Data Analysis. Reading, Mass.: Addison-Wesley.

Wang, J., Li, G., Kraska, T., Franklin, M. J., & Feng, J.(2013). Leveraging transitive relations for crowdsourced joins. *SIGMOD Conference 2013*, 229-240.

Wilson, R. J.(2010). Introduction to Graph Theory (5th ed.), Prentice Hall.

8 Table of Figures

9 Table of Tables