Pay-less Entity Consolidation – Exploiting Entity Search User Feedbacks for Pay-as-you-go Entity Data Integration

Thanh Tran¹
duc.tran@kit.edu

Yongtao Ma¹
ma.yongtao@aifb.uni-karlsruhe.de

Gong Cheng²
gcheng@nju.edu.cn

¹ Institute AIFB, Karlsruhe Institute of Technology, D-76131 Karlsruhe, Germany
² State Key Laboratory for Novel Software Technology, Nanjing University, Nanjing 210093, China

ABSTRACT
Linked Data consists of billions of RDF triples from hundreds of different sources on the Web. The effective construction and maintenance of links between these sources largely depend on data integration solutions that scale to the large volume and heterogeneity of the Linked Data Web. In this context, a promising direction is the pay-as-you-go paradigm that advocates the use of user feedback for an interactive and incremental approach to data integration—to obtain a solution that continuously improves as the underlying system evolves. In this paper, we study pay-as-you-go data integration in the context of entity search. Compared with asking users to contribute to the result quality without to directly benefit from their effort, we show that users “pay less” when entity consolidation is inherently embedded in entity search. In this setting, users interact with the system for solving their tasks, and as a side-effect, contribute to the quality of the consolidation results. We propose an iterative clustering procedure to implement this concept of pay-less entity consolidation. We demonstrate its promising advantages over traditional solutions grounded on an extensive evaluation.

Author Keywords
Clustering, entity consolidation, entity search, implicit user feedback, pay-as-you-go data integration

ACM Classification Keywords
D.2.12 Software Engineering: Interoperability—data mapping; H.3.3 Information Storage and Retrieval: Information Search and Retrieval—clustering, information filtering, relevance feedback, search process, selection process

INTRODUCTION
The amount of semantic data on the Web today is large and increasing. For instance, the amount of Linked Data is in the order of billions of RDF triples from hundreds of sources. Naturally, the increased value of Linked Data comes from the links and accordingly, the construction and maintenance of links have become a hot research topic in the Linked Data community.

Among all possible links that can be established between sources, the types of links that are of particular interest to our community are “sameAs” links at the level of individual resources as well as the links at the schema level. The construction of these links can be greatly facilitated using (semi-)automatic tools. For building these tools, we can leverage the large body of work on data integration at the instance level known as entity consolidation, record linkage [6], etc., and the work at the schema level [18]. Research in the Semantic Web community so far focused on the task of ontology matching [7], but due to the Linked Data trend, also the problem of instance-level integration has attracted interests [13, 25]. However, whereas existing data integration approaches have been studied in and successfully applied only to controlled enterprise settings, the management of links in Linked Data requires solutions that scale to the volume and heterogeneity of the Web.

Recently, large-scale data integration approaches studied in the setting of dataspace follow the pay-as-you-go paradigm [9, 19]. In this setting where we have a large number of dataspaces containing data in different domains, it is no longer affordable to invest an enormous amount of effort in one-time upfront data integration. To deal with this, pay-as-you-go data integration advocates an incremental approach. Instead of relying on a completely integrated system, the idea is to continuously improve the degree and quality of integration as the system evolves. At any time, the system supports users with the best effort, i.e. using the highest quality of integration that is available. In particular, this paradigm emphasizes the role of user involvement. Result quality comes not only from the matching algorithm but also from user feedback and the way it is exploited. However, the pay-as-you-go concept is rather new and only little work can be found in this direction. In particular, it remains an open
question as *how to effectively involve users*.

Observing that user feedback is clearly beneficial and possibly essential to the large-scale and heterogeneous Linked Data setting, this work studies the problem of user involvement in the pay-as-you-go context, focusing on the task of entity consolidation. The main contributions can be summarized as follows:

- In a systematic fashion (abstracting from the algorithms and techniques used for data integration), we classify *user feedback needed* for improving the result quality.

- Instead of putting users directly into the integration task where they have to contribute to the result quality but not directly benefit from their effort [14, 19], we focus on the problem of user involvement in entity consolidation and show that users *pay less* when consolidation is inherently embedded in entity search. In this setting, users interact with the system for solving their tasks (i.e. searching for an entity), and as a side-effect, contribute to the quality of the consolidation results.

- For implementing this “pay-less” concept of entity consolidation in entity search, we propose an *iterative clustering procedure containing a pipeline of operations*. Every operation either is performed by the system or represents a user activity. During this process, the combination of machine and human involvement helps to iteratively improve the consolidation and the search results such that both the machine and the user benefit from their effort.

- In our experiments comprising real-world data sets, queries, and users, we show that much more complete and exact consolidated information is achieved by users through their affordable effort in our pay-less setting than in a fully-automatic way. Meanwhile, users also effectively contribute to help the system improve consolidation via the proposed iterative clustering procedure.

In the remainder of this paper, firstly we discuss user involvement in pay-as-you-go data integration. Then we propose our pay-less consolidation procedure in the context of entity search, describe an implementation, and present evaluation results. Finally we compare related work and conclude the paper with future work.

**USER INVOLVEMENT IN PAY-AS-YOU-GO DATA INTEGRATION**

Pay-as-you-go and particularly the involvement of users are important in the Linked Data Web environment because the given scale and heterogeneity of data challenge the applicability of the work that has been proposed so far. Incremental data integration that is performed only on some subsets by the help of users can solve both the problem of scalability and quality.

The main goal of data integration is to find data elements (or classes/properties of elements) that refer to the same real-world object (or class/property of objects). A pair of such elements is commonly defined as a mapping:

\[
\text{DEFINITION 1 (MAPPING). A mapping } m(e_i, e_j) \text{ comprises an unordered pair of data elements } e_i \text{ and } e_j \text{, and is called positive if } e_i \text{ and } e_j \text{ refer to the same real-world object, otherwise is called negative.}
\]

Computing these pairs at the schema level result in *class/property mappings*, whereas in entity consolidation the goal is to compute mappings between instances (RDF resources that are not classes/properties in the Linked Data context), also called *entity mappings*.

Generally speaking, computing these mappings requires the system to identify appropriate features to characterize the data elements and to measure the similarities between these elements. Simple algorithms based on distance measures up to sophisticated machine learning approaches that learn features, feature weights, as well as the similarity function and its parameters have been employed [4, 5, 16]. With respect to this kind of integration result to be expected and the way it is computed, user involvement can take place at different levels:

- Users can contribute *at the level of features*. In fact, this is commonly the case in enterprise integration scenarios where domain experts are employed to specify or revise (combination of) features.

- Users can contribute *at the level of results*. Users can manually create mappings. They can also refine automatically computed mappings by confirming positive and rejecting negative ones. Since there may be too many candidate mappings that require this kind of user feedback, a more efficient way is to identify and focus on the ones that provide the largest benefit to the system in terms of data integration quality [14, 24].

- Users can contribute by providing different hints for the system to *learn faster*. This kind of contribution is actively used in the field of active learning where the types of contributions needed vary with the details of the underlying techniques [23].

Unfortunately, contributions of any types require additional effort for which somehow has to be compensated to keep users investing. Providing incentives for users to continuously contribute is essential to pay-as-you-go integration, and the lack of these incentives so far hindered the success of this paradigm.

**PAY-LESS CONSOLIDATION IN ENTITY SEARCH**

In this section, we present our work on pay-as-you-go entity consolidation in the Linked Data setting. We incorporate the consolidation task into the entity search process so that users implicitly contribute to consolidation when they improve search results, and we call it pay-less consolidation. In the following, we discuss entity search, introduce how to embed consolidation into this process, and present an iterative procedure for reasons of scalability and usability.

**Entity Search in the Pay-as-you-go Setting**
Different from the traditional Web environment where entity descriptions are embedded in textual data, Linked Data contains descriptions that are richer in structure and semantics, leveraging RDF as the data model for capturing entities’ attribute values and relationships. In RDF, an entity is identified by a URI, and is connected to literals and other entities via links (i.e., properties). A link and its target form a property-value pair (i.e., an RDF triple), and based on this, the description of an entity can be characterized as a set of property-value pairs found for a given URI (i.e., a set of RDF triples which have the same URI in the first position, also called the subject).

Entity search is a task where the goal is to find one particular entity and to obtain its description. This task is commonly supported by Semantic Web search engines such as Falcons [2] and SWSE [12]. This task is related to the well-known concept of navigational search [1], which is about finding one particular Web page. Considering a Web page as a particular type of entity description, entity search can be regarded as a generalization of this concept. Finding an entity or a Web page is not always an end goal but also, might be regarded as a generalization of this concept. Finding an entity or a Web page is not always an end goal but also, might be performed to obtain a starting point from which the user continues with additional search and browsing. Thus, the entity to be searched for is mostly known such that the user knows how to formulate a corresponding query. An analysis of Web query log [22] reveals that entity search is very popular, largely dominating all other types of search such as attribute search or relationship search.

At present, mainstream Web and Semantic Web search engines (e.g., Google, Yahoo!, Falcons, SWSE) accept keyword queries as the primary type of input. Following this line, this work also considers entity search with keyword queries.

The problem of entity search in the pay-as-you-go context is that as we do not fully assume a completely integrated system, several descriptions (for different URIs) that actually refer to the same real-world object can be obtained as results for a given query. These descriptions might contain complementary information, which when combined, constitutes a more complete description of the entity. In order to obtain more complete results, the user might have to go through the list of results and manually merge them. By incorporating entity consolidation into this process, we aim at helping users obtain more complete results, whereas also leveraging their implicit feedback to improve the quality of consolidation.

Manual Consolidation Based on Implicit Feedback

We have discussed the different ways users can explicitly contribute to improve integration results. However, creating and verifying mappings for instance, are far from the tasks users hope to accomplish with entity search. As discussed, the primary task is to find information for a particular entity. We observe that for this purpose, the user actually has to go through the list of results to identify those entity descriptions that contain information relevant to the entity in question. Since all computed results in the list are supposed to refer to the same entity, by selecting relevant results from the list, the user actually confirms positive mappings and indirectly, rejects negative ones. As a result, we have two sets of entity descriptions:

- \( E^R \subseteq E \) containing relevant entities which refer to one and the same real-world object the user searches for, and
- \( E^{IR} = E \setminus E^R \) containing irrelevant entities because they refer to other object(s).

Then \( E^R \) are consolidated and their descriptions are combined to produce a more complete description. Thus, whereas users interact with the system to improve search results, they indirectly perform entity consolidation. Since these user activities primarily aim at the search task, they are considered as implicit feedback and the process amounts to manual entity consolidation.

Note that the results of this classification are two clusters, which capture some entity mappings. As illustrated in Fig. 1 and defined below, we obtain both positive and negative mappings.

**Definition 2 (Mappings Derived from Entity Search).**

Given \( E^R \) and \( E^{IR} \),

- for all \( e_i, e_j \in E^R \), \( m(e_i, e_j) \) forms a positive entity mapping, and
- for all \( e_i \in E^R, e_j \in E^{IR} \), \( m(e_i, e_j) \) forms a negative entity mapping.

That is, through this entity search process, the user contributes \( \binom{|E^R|}{2} \) positive entity mappings and \( |E^R| \cdot |E^{IR}| \) negative entity mappings to the system. Not only the user can benefit from this effort in terms of more complete results, but also these mappings can be leveraged by the system to improve for future search requests, i.e., to continuously improve the integration quality as users employ the system for their daily tasks.

**Semi-automatic Clustering-based Consolidation**

In real-world settings, an entity search engine might return a large set of entities \( E \) for a given keyword query. Classifying \( E \) one by one to manually produce these two clusters is rather inefficient. Hence we propose to group similar entities and let users verify groups of entities instead. The workflow is illustrated in Fig. 2.

To this end, firstly the system carries out a clustering operation on the results (entities), by using features extracted from their descriptions.

**Definition 3 (Clustering).** Given a set of entities \( E_i \) and a positive integer \( k \), the results of clustering \( E_i \) is a partition of \( E_i \) denoted by \( \text{Clustering}_{k}(E_i) = \{ E_{i,1}, E_{i,2}, ..., E_{i,k} \} \) where \( |\text{Clustering}_{k}(E_i)| \) is \( k \) or less.

Since a cluster might contain a large number of entities, a summary of the combined set of corresponding entity de-
m <AIFB_Studer, R.Studer> = positive  

m <Rudi_Studer, C.Studer> = negative

Figure 1. Entity mappings derived from users’ confirmation of entity search results.

Figure 2. Semi-automatic clustering-based consolidation (one-time).

Descriptions is computed to present only the features that are most important to understand the content of the cluster and to distinguish the cluster from other clusters. To this end, the selected features are expected to be shared commonly by the entities within the cluster, but are rarely observed outside the cluster [17].

Finally, the user identifies relevant clusters from $\text{Clustering}_k(E_i)$, the union of which will form $E_i^R$. Similar to selecting entities, we expect the user to choose those clusters that contain the entity being searched for. All the selected clusters are then combined to form one single cluster, i.e. the one containing the entity descriptions the user looks for. Thus, this operation is a selection, and at the same time, a confirmation indicating that all the entities in the selected clusters refer to the same real-world object:

**Definition 4 (Select-and-Confirm).** From $\text{Clustering}_k(E_i)$, the user selects (several) cluster(s) $E_{i,s} \in \text{Clustering}_k(E_i)$, which results in $E_i^{R} = \bigcup_s E_{i,s}$.

Note that multiple selections are allowed since a clustering operation performed by the system may mistakenly put relevant entities into different clusters.

Iterative Semi-automatic Clustering-based Consolidation

Clearly, not all the entities in a cluster necessarily refer to the same real-world object. That is, a cluster might contain negative mappings. Further, when there are an extremely large number of entities to be classified, the true value of $k$ could be so large that users are required to go through many clusters. For these reasons, besides the one-time select-and-confirm operation, we propose a procedure where users can iteratively refine the cluster obtained in every step.

In particular, we use a manageable value of $k$ so that all results can be inspected quickly. After the user has made the selections, the union cluster resulted from this is not always the final result as in the previous workflow, but could be submitted again to the system for further refinement. In that next iteration, the system again performs clustering, and the user makes the selections. When the user does not choose all the results, every refinement results in more fine-grained clusters. This iterative process goes on until the user chooses to carry out a select-and-confirm operation. This new workflow is illustrated in Fig. 3.

To support this iterative procedure, a select-and-refine operation is needed:

**Definition 5 (Select-and-Refine).** From $\text{Clustering}_k(E_i)$, the user selects (several) cluster(s) $E_{i,s} \in \text{Clustering}_k(E_i)$, which results in $E_{i+1} = \bigcup_s E_{i,s}$ to be used in the next iteration.
IMPLEMENTATION
Here we present an implementation of the proposed procedure. The system mainly consists of three components: keyword search, clustering, and result presentation. Clearly, there exist a large number of approaches proposed for dealing with each of these three problems [2, 8, 11, 12, 17, 26].

For keyword search, we employ the technique used by standard Semantic Web search engines [2, 26]. Firstly, we construct a virtual document for each entity that consists of all the terms found in its description. Then, an inverted index, a data structure widely used in information retrieval (IR), is built for indexing these virtual documents. Given a keyword query, this index can be used in the standard IR-fashion to retrieve matching documents, i.e., entities in this case.

For clustering, we define a feature as a property-value pair taken from an entity description. Thus every entity description constitutes a feature set. The similarity between two entities is given by the Jaccard coefficient between their feature sets. For efficient computation, we use a specialized index for retrieving entities’ feature sets. Based on this, the Hierarchical Agglomerative Clustering (HAC) algorithm [20] is implemented, which initially treats each entity as a singleton cluster and then successively merges pairs of clusters until the number of the remaining clusters is not larger than a given value of \( k \). Specifically, we choose the average-link distance measure so that the algorithm always merges the pair of clusters with the highest cohesion, i.e., we use the average of the similarities between all pairs of entities across clusters.

To compute a summary for a cluster, we collect property-value pairs of all its constituent entities and for each, we extract terms from their local names (in the case of URI) or lexical forms (in the case of literal) and represent them as a term vector weighted using Term Frequency-Inverse Document Frequency (TF-IDF). Then given a keyword query, top-ranked property-value pairs based on the cosine similarity between the term vector of the query and the term vector of a property-value pair constitute a summary.

EVALUATION
In the evaluation we study the following two aspects: how the quality of entity consolidation is improved via pay-less user involvement, and to what extent users benefit from their effort w.r.t. their entity search tasks.

Design
We provide two modes of interaction for users to interact with entity search results. In the manual mode, the top-100 entities returned by keyword search are presented. We treat each entity as a singleton cluster and provide \( m \) of its property-value pairs as a summary of its description. By examining these summaries one by one, users identify relevant entities. In the semi-automatic mode, we implement the proposed one-time and iterative clustering procedures. Also starting with the top-100 entities, (remaining) entities are grouped into \( k \) clusters in each iteration. A summary consisting of \( m \) property-value pairs is presented, based on which users select relevant clusters and then continue to refine or confirm. Altogether, we study 4 different approaches: (1) fully-manual, (2) one-time clustering, (3) iterative clustering, and (4) fully-automatic. Results for one-time clustering are not separately computed, but simply derived from the iterative one, i.e., the results obtained after the first iteration. The fully-automatic one is the same as one-time clustering except that the system takes each cluster as the results at a time instead of referring to user selection.

We use the first 10 chunks of the Billion Triples Challenge 2010 data set (containing 100M RDF triples). From the queries proposed for the Entity Search Evaluation Benchmark at SemSearch 2010,\(^1\) we select 20 against which our

\(^{1}\)http://km.aifb.uni-karlsruhe.de/ws/semsearch10/Files/finalqueries.
data set returns the highest number of results. We use different settings for k and m, i.e., manual mode with m = 10, semi-automatic mode with fixed m = 10 and k ∈ {5, 10, 20}, and semi-automatic mode with fixed k = 5 and m ∈ {5, 10, 15}. This means a query has to be run 6 times (called tasks). Further, we require every task to be performed 3 times by 3 different users, resulting in a total of 360 tasks (20 × 6 × 3). We have 12 student participants. After 2 warm-up tasks (one for each mode) that are not recorded, every participant conducts 30 tasks.

Evaluating Entity Consolidation Results

Firstly we aim to assess the quality of entity consolidation results based on the mappings returned by each approach. In this case, mappings are captured by the set of relevant entities returned by each approach, which shall denote the same real-world object being searched for. We use the full-manual approach to establish the gold standard. Since there are always 3 user judgments for every query, an entity is considered relevant and added to the gold standard (E_R) only if it is selected by at least 2 users. However, even if entity search returns relevant entities indeed, users sometimes cannot identify them, mainly because of the low quality of data, i.e., entity descriptions often contain such few property-value pairs that they can hardly be correctly identified without further exploring related entities and sources (which are not part of the search results). As a result, the gold standard could not be established for 3 queries, i.e. E_R is empty, and thus, they are excluded from the following evaluation results.

Then given a set of relevant entities E_R returned by approaches (2)–(4), we compute recall via \( \frac{|E_R \cap E_{\text{gold}}|}{|E_{\text{gold}}|} \) and precision via \( \frac{|E_R \cap E_{\text{gold}}|}{|E_R|} \). We also measure efficiency in terms of the time spent from the generation of the top-100 entities to the final confirmation (which is not applicable to approach (4)), which mostly represents the time for user interaction because the time for clustering is relatively negligible. Figure 4 shows the average results computed over all tasks for different settings.

The fully-automatic approach achieves very low recall in the range of 0.05–0.20 (i.e. for different parameter settings) and also very low precision in the range of 0.03–0.04. Through one-time user involvement, recall improves and is in the range of 0.25–0.45. For different parameter settings, iterative clustering further increases precision, which is in the range of 0.07–0.31. However, whereas precision is consistently better than one-time clustering (0.04–0.24 improvement), recall is worse (0.11–0.23 reduction). With respect to time performance, we can conclude that clustering-based approaches require less effort. Taking the time needed for the manual mode as the basis, times for one-time clustering and for iterative clustering make up 17%–39% and 44%–74%, respectively.

As shown in Fig. 4(a), precision and recall increase with larger value of k. For iterative clustering, precision increases by 0.24 and recall increases by 0.16 when increasing k from 5 to 20, whereas time increases by only 15%. Time even surprisingly decreases from k = 10 to k = 20 when using one-time clustering. It seems that a larger number of clusters and thus fewer entities in a cluster make it easier for users to understand and to identify relevant descriptions.

To investigate the impact of m, as shown in Fig. 4(b), although the time increases by around 60% from m = 5 to m = 15, precision and recall do not increase significantly or consistently. Thus, short summaries seem to be effective in trading a small decrease in result quality for a large increase in efficiency.

Given that the BTC data set is representative of real-world Web data, and given that it is of low quality, user involvement seems to be essential. In fact, even with the help of users in the manual mode, no perfect results could be achieved because the data provides only limited information for users to make decisions. The fully-automatic approach performs poorly on this Web data. These results could be substantially improved via user involvement. Compared to one-time involvement, the proposed iterative approach shows that a large increase in precision can be achieved at the cost of a smaller decrease in recall. This is important for the pay-as-you-go setting where recall (i.e. completeness of integration) is secondary and obtaining more precise results via user feedback is the main goal. We conclude from this experiment that pay-as-you-go with (iterative) user feedback is promising—and might be needed for improving not only consolidation quality but also the quality of the underlying data (as the user exploits it).

Besides, we are aware of that although the proposed approach has performed relatively much better, the absolute precision and recall values achieved are still unsatisfactory. This is partially due to the simple similarity function and clustering algorithm adopted in the implementation. As mentioned, these topics are out of the scope of this paper, but more advanced solutions are to be tested in future work.

Evaluating Entity Search Results

Now we study how much users benefit from the feedback they implicitly provide to the system during entity search. The goal of the search is to obtain information about the entity in question. Thus, users benefit when the information is more complete and exact. We propose to measure these aspects in terms of the number of property-value pairs that can be obtained after consolidation. For each query, let P^R be the set of property-value pairs extracted from the descriptions of relevant entities in the gold standard (i.e. E_R in the previous experiment). Then given a set of property-value pairs P^R derived from the descriptions of entities in clusters obtained via consolidation using approaches (2)–(4), we compute its completeness as \( \frac{|P^R \cap P_G^R|}{|P_G^R|} \) and exactness as \( \frac{|P^R \cap P_G^R|}{|P^R|} \). Efficiency measured in time is the same as the previous experiment. Figure 5 shows these results in different settings, being averaged over all tasks.

\[ \text{http://ws.nju.edu.cn/falcons/export/20_queries_from_semsearch2010.txt.} \]
Compared with the fully-automatic approach, the completeness of consolidated information in the semi-automatic mode with one-time clustering increases by 0.01–0.39 (depending on the parameter setting), whereas the exactness is similar. This is achieved at the cost of 41–91 seconds. With iterative clustering and $k = 20$ for instance, not only the completeness increases by 0.22 but also the exactness increases by 0.26, at the cost of 175 seconds. However, benefiting from the iterative process, users can choose their own trade-offs between the quality of consolidated information obtained and the time spent.

As shown in Fig. 5(a), both completeness and exactness are significantly improved as $k$ increasing from 5 to 20, whereas the additional amount of time needed is rather small. Thus, a relatively large value of $k$ seems to be favorable.

Fig. 5(b) shows the effect of $m$. Increasing $m$ from 5 to 10 and 15 requires 48%–63% additional time but also results in noticeable increases in both completeness (0.03–0.12) and exactness (−0.01–0.06).

In summary, the experiments have shown that by providing feedback to the system, users not only improve consolidation results but also benefit from more complete and exact entity search results. However, whereas user involvement is critical to pay-as-you-go consolidation, also the tools require improvement. We find there is large potential for improving current results, and the problem here lies not only in the low-quality data but also in the low-quality summaries derived from that, providing very limited means for users to quickly decide which entities and clusters are relevant. This calls for future work on approaches for summarizing entity descriptions, and concepts employed by approaches for ontology summarization [28] provide theoretical underpinnings to start from.
RELATED WORK

We now elaborate on related work already mentioned throughout the paper.

Entity Consolidation

Entity consolidation has been studied extensively in the literature on various data models such as relational [6], XML [15], and Web ontologies [7]. A paradigm widely adopted is to characterize an entity as a set of features and then calculate the similarity between two entities by inspecting the overlap between their feature sets. The feature set of an entity can be defined in many ways, e.g. in terms of concepts (i.e. classes of the entity) [8], in terms of neighboring entities in the RDF graph [10], as property-value pairs [13], or as paths in the RDF graph starting from the entity [11]. Whereas fully-automatic approaches have shown to perform well in enterprise scenarios, the use of human attention has attracted interests. Entity mappings provided by human experts have been leveraged to train a machine learning approach for consolidation [5], to determine the weights assigned to different consolidation strategies [4], and to filter mapping suggestions [16]. However, incentives and motivations for users to actively create and refine entity mappings are still missing.

Entity Search

Web search is shifting from document search to the generalized notion of entity search employed here [3, 21]. Linked Data suits this trend well since it provides structured entity descriptions that can be effectively used for search [2, 12]. However, very little work has been invested in dealing with the problem of heterogeneity in this new environment. Most results returned by current systems have to be consolidated, demanding for a pay-as-you-go solution. A pioneer in this field is Sig.ma [27], which consolidates entities using simple heuristics, and presents users with a list of sources that can be manually refined.

Pay-as-you-go Data Integration

Pay-as-you-go data integration is proposed for dealing with dataspaces [9] and has been implemented at Google [19]. It treats integration as an ongoing process that starts with disparate data from different sources and incrementally improves integration over time. This paradigm aims at combining automatic algorithms for data integration with user feedback. Since there are usually too many candidate mappings that could benefit from user feedback, recent work in this direction primarily considers the problem of finding a subset of mappings that offer the highest utility to the system [14], which is related to the problem of active learning [23, 24]. However, all these approaches are based on explicit feedback, i.e. interrupting users with requests not related to their daily tasks.

Comparison

The aim of our work is not to study a new fully-automatic algorithm for entity consolidation, but is to employ user feedback in a pay-as-you-go fashion. Different from previous work that explicitly involves users in a consolidation task, our approach on the one hand lets users interact with the system for solving their entity search tasks, and on the other hand implicitly collects their contributions and uses them to improve the quality of consolidation results. As a result, users not only pay less since entity consolidation is inherently embedded in applications (e.g. entity search), but also directly benefit from their contributions (e.g. obtaining more accurate results). This gives incentives for users to continuously contribute as the system evolves. Rather than asking users to refine the results one by one, we leverage clustering techniques to group the results so that the refinement can be more efficiently performed in a batch mode. Further, this process is extended to be performed in an iterative fashion.

CONCLUSIONS AND FUTURE WORK

In this paper, we have proposed the concept of pay-less entity consolidation and implemented it in the context of entity search on Linked Data. This pay-less concept follows the pay-as-you-go paradigm that combines automatic algorithms for consolidation with user feedback to improve consolidation results incrementally over time and effort. Instead of interrupting users with explicit feedback requests, we embed this consolidation task inherently in the search process so that users contribute in an implicit way, i.e., while performing entity search, users actually provide feedback to improve consolidation results. By means of an iterative clustering procedure consisting of a pipeline of operations carried out by the system and user, we argue that not only consolidation but also entity search results can be improved. Thus, users pay less in the sense that they benefit from their effort. This incentive inherently available in the system is important to ensure continuous user involvement and feedback as the system evolves.

Based on an extensive evaluation with real-world data and queries, we have demonstrated the value of this pay-less concept. We find real-world data is of low quality and an automatic approach yields very low-quality results, which could be improved substantially via our pay-less user involvement concept. However, we also see large potential for improving tools. For instance, not only the data but also the summaries derived from it provide only limited means for users to provide the right feedback. This summarization problem has not been addressed extensively in the literature, and is regarded as the primary direction of future work towards a more effective implementation of our iterative procedure. As to user feedback, in this work we have only discussed how to collect it in an inexpensive way; in future work, we will investigate how to leverage it to improve consolidation in an appropriate way, e.g. how to resolve potential disagreements on mappings between different users.

Acknowledgments.

This work was supported by the NSFC under Grant 61100040 and 61003018. We would like to thank all the participants in
the experiments.

REFERENCES


