Record Deduplication using Sporogenous Programming Approach

Gousiya Begum, S. Raja Manohara Rao, K. Mona Chary

Computer Science and Engineering Department, MGIT, Hyderabad
INDIA

Abstract — Different types of digital libraries and e-commerce websites are exist with duplicate contents. Previously many systems are present for removing replica or duplicate items. Previous approaches are implemented in different repositories for detection of duplicate records. It can provides the organized or alignment based results. Those approaches are detect the results are near duplicate and range based results. These approaches are reducing the computation cost and time. Result is not contains any quality data.

Increasing the digital libraries data quality new approaches are implementing in present system. This new approach is known as a sporogenous programming popularly known as sporogenous programming. Sporogenous programming contains the three major operations here. Those operations are selection, crossover and mutation. All three operations are performing in database. Execution of operations applies the de-duplication function. After removing the duplicate records apply the suggested function. These two functions are working properly and it can contain less computational resources utilization in implementation. Compare to all previous approaches present approach provides less burden, efficient and accurate results display here. It can provide good evidence based results.

Keywords: sporogenous programming, data base integration, sporogenous algorithms.

I. INTRODUCTION

The increasing volume of information available in digital media has become a challenging problem for data administrators. Usually built on data gathered from different sources, data repositories such as those used by digital libraries and e-commerce brokers may present records with disparate structure. Also, problems regarding low-response time, availability, security, and quality assurance become more difficult to handle as the amount of data gets larger. It is possible to say that the capacity of an organization to provide useful services to its users is proportional to the quality of the data handled by its systems. In this environment, the decision of keeping repositories with “dirty” data goes far beyond technical questions such as the overall speed or performance of data management systems. The solutions available for addressing this problem requires more than technical efforts, they need management and cultural changes as well [1]. To better understand the impact of this problem, it is important to list and analyse the major consequences of allowing the existence of “dirty” data in the repositories. These include, for example:

1) Performance degradation—as additional useless data demand more processing, more time is required to answer simple user queries;
2) Quality loss—the presence of replicas and other inconsistencies leads to distortions in reports and misleading conclusions based on the existing data;
3) Increasing operational costs—because of the additional volume of useless data, investments are required on more storage media and extra computational processing power to keep the response time levels acceptable.

To avoid these problems, it is necessary to study the causes of “dirty” data in repositories. A major cause is the presence of duplicates, quasi replicas, or near-duplicates in these repositories, mainly those constructed by the aggregation or integration of distinct data sources. The problem of detecting and removing duplicate entries in a repository is generally known as record deduplication [2].

The main contribution of this paper is a GP-based approach to record deduplication that:
1) Outperforms an existing state-of-the-art machine learning based method found in the literature;
2) Provides solutions less computationally intensive, since it suggests deduplication functions that use the available evidence more efficiently;
3) Frees the user from the burden of choosing how to combine similarity functions and repository attributes. This distinguishes our approach from all existing methods, since they require user-provided settings;
4) Frees the user from the burden of choosing the replica identification boundary value, since it is able to automatically select the deduplication functions that better fit this deduplication parameter.

II. LITERATURE SURVEY

Record deduplication is a growing research topic in database and related fields such as digital libraries. Today, this problem arises mainly when data are collected from disparate
sources using different information description styles and metadata standards. Other common place for replicas is found in data repositories created from OCR documents. These situations can lead to inconsistencies that may affect many systems such as those that depend on searching and mining tasks. To solve these inconsistencies it is necessary to design a deduplication function that combines the information available in the data repositories in order to identify whether a pair of record entries refers to the same real-world entity.

In the realm of bibliographic citations, for instance, this problem was extensively discussed by Lawrence et al [3], [4]. They propose a number of algorithms for matching citations from different sources based on edit distance, word matching, phrase matching, and subfield extraction.

a) RELATED LITERATURE:

As more strategies for extracting disparate pieces of evidence become available, many works have proposed new distinct approaches to combine and use them. Elmagarmid et al. [5] classify these approaches into the following two categories: 1) Ad-Hoc or Domain Knowledge Approaches—this category includes approaches that usually depend on specific domain knowledge or specific string distance metrics. Techniques that make use of declarative languages can be also classified in this category; 2) Training-based Approaches—This category includes all approaches that depend on some sort of training—supervised or semi-supervised—in order to identify the replicas. Probabilistic and machine learning approaches fall into this category.

The idea of combining evidence to identify replicas has pushed the data management research community to look for methods that could benefit from domain-specific information found in the actual data as well as for methods based on general similarity metrics that could be adapted to specific domains. As an example of a method that exploits general similarity functions adapted to a specific domain, we can mention.

There the authors propose a matching algorithm that, given a record in a file (or repository), looks for another record in a reference file that matches the first record according to a given similarity function. The matched reference records are selected based on a user-defined minimum similarity threshold. Thus, more than one candidate record may be returned. In such cases, the user is required to choose one among them indicating which the closest one is. Records matching on high-weight tokens (strings) are more similar than those matching on low-weight tokens. The weights are calculated by the well-known IDF weighting method.

This weighting method is also used in WHIRL, a database management system that supports similarity joins among relations that have free text attribute values. In, the authors use the vector space model for computing similarity among fields from different sources and evaluate four distinct strategies to assigning weights and combining the similarity scores of each field. As a result of their experiment, they found that using evidence extracted from individual attributes improves the results of the replica identification task.

Newcombe et al [6], were the first ones to address the record deduplication problem as a Bayesian inference problem (a probabilistic problem) and proposed the first approach to automatically handle replicas. However, their approach was considered empirical since it lacks amore elaborated statistical ground.

After Newcombe et al.’s work, Fellegi and Sunter [7] proposed a more elaborated statistical approach to deal with the problem of combining evidence. Their method relies on the definition of two boundary values that are used to classify a pair of records as being replicas or not.

Tools that implement this method, such as FebRI, usually work with two boundaries as follows:
1. Positive identification boundary—if the similarity value lies above this boundary, the records are considered as replicas;
2. Negative identification boundary—if the similarity value lies below this boundary, the records are considered as not being replicas.

For the situation in which similarity values stand between the two boundaries, the records are classified as “possible matches” and, in this case, a human judgment is necessary.

Usually, most of the existing approaches to replica identification depend on several choices to set their parameters, and they may not be always optimal. Setting these parameters requires the accomplishment of the following tasks: 1) Choosing the best evidence to use—the more evidence, the more time is required to find the replicas, since more processing is needed to calculate the similarity among the attributes.
2) Finding how to combine the best evidence—some evidence may be more effective for replica identification than others;
3) Finding the best boundary values to be used—bad boundaries may increase the number of identification errors (e.g., false positives and false negatives), nullifying the whole process.

The proposals that are more related to our work are those that apply machine learning techniques for deriving record level similarity functions that combine field-level similarity functions, including the proper assignment of weights [8], [9], [10], [11]. These proposals use a small portion of the available data for training. This training data set is assumed to have similar characteristics to those of the test data set, which makes feasible to the machine learning techniques to generalize their solutions to unseen data. The good results usually obtained with these techniques have empirically demonstrated that those assumptions are valid. In [8] and [9], the authors use a machine learning technique to improve both the similarity functions that are applied to compare record fields and the way the pieces of evidence are combined. In their system, called Marlin, the extracted evidence is encoded as feature vectors, that are used to train a Support Vector Machine classifier to better combine them in order to identify replicas. The main idea behind this approach is that, given a set of record pairs, the similarity between two
attributes (e.g., two text strings) is the probability of finding the score of best alignment between them, so the higher the probability, the bigger the similarity between these attributes. They compare the Marlin system with several other effective learning methods and show it outperforms all of them. The Marlin system is used here as our baseline since it is the current state-of-the-art method for the record deduplication problem. The adaptive approach presented in [10] consists of using examples for training a learning algorithm to evaluate the similarity between two given names, i.e., strings

III. SYSTEM DESIGN

a) Sporogenous programming basic concept:

Evolutionary programming is based on ideas inspired on the naturally observed process that influence virtually all living beings, the natural selection. Sporogenous Programming is one of the best known evolutionary programming techniques. It can be seen as an adaptive heuristic whose basic ideas come from the properties of the sporogenous operations and natural selection system. It is a direct evolution of programs or algorithms used for the purpose of inductive learning (supervised learning), initially applied to optimization problems. GP, as well as other evolutionary techniques, is also known for its capability of working with multi objective problems that are normally modeled as environment restrictions during the evolutionary process. GP and other evolutionary approaches are also widely known for their good performance on searching over very large—possibly infinite—search spaces, where the optimal solution in many cases is not known, usually providing near-optimal answers. As stated by Koza, “the fact that the GP algorithm operates on a population of individuals, rather than on a single point in the search space of the problem, is an essential aspect of the algorithm.” This can be explained since the population serves as the pool of the probably valuable sporogenous material, which is used to create new solutions with probably valuable new combinations of features.

The main aspect that distinguishes GP from other evolutionary techniques (e.g., sporogenous algorithms, evolutionary systems, sporogenous classifier systems) is that it represents the concepts and the interpretation of a problem as a computer program—and even the data are viewed and manipulated in this way. This special characteristic enables GP to model any other machine learning representation.

b) Sporogenous Operations:

Usually, GP evolves a population of length-free data structures, also called individuals, each one representing a single solution to a given problem. In our modelling, the trees represent arithmetic functions, as illustrated in Figure 1.

![Figure 1 selection](image)

When using this tree representation in a GP-based method set of terminals and functions should be defined.

Terminals are inputs, constants or zero argument nodes that terminate a branch of a tree. They are also called tree leaves. The function set is the collection of operators, statements, and basic or user-defined functions that can be used by the GP evolutionary process to manipulate the terminal values. These functions are placed in the internal nodes of the tree, as illustrated in Fig.3.1. During the evolutionary process, the individuals are handled and modified by sporogenous operations such as reproduction, crossover, and mutation, in an iterative way that is expected to spawn better individuals (solutions to the proposed problem) in the subsequent generations. Reproduction is the operation that copies individuals without modifying them. Usually, this operator is used to implement an elitist strategy that is adopted to keep the sporogenous code of the fittest individuals across the changes in the generations. If a good individual is found in earlier generations, it will not be lost during the evolutionary process. The crossover operation allows sporogenous content (e.g subtrees) exchange between two parents, in a process that can generate two or more children. In a GP evolutionary process, two parent trees are selected according to a matching (or pairing) policy and, then, a random sub tree is selected in each parent. Child trees are the result from the swap of the selected sub trees between the parents, as illustrated in Figure 2.

![Figure 2 mutation](image)

Finally, the mutation operation has the role of keeping a minimum diversity level of individuals in the population, thus avoiding premature convergence. Every solution tree resulting from the crossover operation has an equal chance of suffering a mutation process. In a GP tree
representation, a random node is selected and the corresponding sub tree is replaced by a new randomly created sub tree, as illustrated in Figure 3.

![Tree Diagram](image)

**Figure 3 Cross over**

All operations for node replacements and insertions performed by mutation and crossover are executed using equal (and constant) probabilities. This way all nodes have the same probability of being chosen in order to guarantee the diversity of the individuals within the sporogenous pool.

### IV. ALGORITHMS USED

There are basically two ways of conducting the evolutionary process: by applying a steady-state algorithm that does not consider distinct generations, or a generational one that does. In this work, the GP evolutionary process is guided by generational evolutionary algorithm. This means that there are well defined and distinct generation cycles. We adopted this approach since it captures the basic idea behind several evolutionary algorithms. The steps of this algorithm are the following:

1) Initialize the population (with random or user provided individuals).
2) Evaluate all individuals in the present population, assigning a numeric rating or fitness value to each one.
3) If the termination criterion is fulfilled, then execute the last step. Otherwise continue.
4) Reproduce the best n individuals into the next generation population.
5) Select m individuals that will compose the next generation with the best parents.
6) Apply the sporogenous operations to all individuals selected. Their offspring will compose the next population. Replace the existing generation by the generated population and go back to Step 2.
7) Present the best individual(s) in the population as the output of the evolutionary process.

### V. EXPERIMENTAL RESULTS AND ANALYSIS

We present and discuss the results of the experiments performed to evaluate our proposed GP based approach to record deduplication. There were three sets of experiments: 1) GP was used to find the best combination function for previously user-selected evidence, i.e., <attribute, similarity function> pair combinations specified by the user for the deduplication task. The use of user selected evidence is a common strategy adopted by all previous record deduplication approaches. Our objective in this set of experiments was to compare the evidence combination suggested by our GP-based approach with that of a state-of-the-art SVM-based solution, the Marlin system, used as our baseline.

2) GP was used to find the best combination function with automatically selected evidence. Our objective in this set of experiments was to discover the impact on the result when GP has the freedom to choose the similarity function that should be used with each specific attribute.

3) GP was tested with different replica identification boundaries. Our objective in this set of experiments was to identify the impact on the resulting deduplication function when different replica identification boundary values are used with our GP based approach.
Identifying and handling replicas is important to guarantee the quality of the information made available by data intensive systems such as digital libraries and e-commerce brokers. These systems rely on consistent data to offer high-quality services, and may be affected by the existence of duplicates, quasi replicas, or near-duplicate entries in their repositories. Thus, for this reason, there have been significant investments from private and government organizations for developing methods for removing replicas from large data repositories. We presented a GP-based approach to record deduplication. Our approach is able to automatically suggest deduplication functions based on evidence present in the data repositories. The suggested functions properly combine the best evidence available in order to identify whether two or more distinct record entries are replicas (i.e., represent the same real-world entity) or not.

We intend to conduct additional research in order to extend the range of use of our GP based approach to record deduplication. For accomplishing this, we plan experiments with data sets from other domains. More specifically, we intend to investigate in which situations (or scenarios) our proposed GP-approach would not be the most adequate to use. Since record deduplication is a very expensive and computationally demanding task, it is important to know in which cases our approach would not be the most suitable option. In addition, we intend to improve the efficiency of the GP training phase by selecting the most representative examples for training. By doing so, we can minimize the training effort required by our GP-based approach without affecting the quality of the final solution.

VII. REFERENCES