الجمـهورية الجزائرية الديمقراطية الشعبيـة

وزارة التعليم العالي والبحث العلمي

جامعة سعيدة د.مولاي الطاهر

كـليـة العلوم

قسم: الإعلام الآلي

Master's Thesis

Specialization: Computer Networks and Distributed Systems

Theme

Integration and Quality Management of Data from Heterogeneous Sources

Presented by:

BEKKOUCHE Yasser

CHAREF Hicham

Supervised by:

Dr. BENYAHIA Miloud

2025-2026

Acknowledgements

First and foremost, we thank Almighty God for giving us the strength, patience, and courage to accomplish this modest work.

We would like to express our deep gratitude to our supervisor, **Mr. BENYAHIA Miloud**, for his guidance, valuable advice, availability, and support throughout the completion of this dissertation.

Our sincere thanks also go to the members of the jury for the honor they have bestowed upon us by accepting to evaluate our work.

We would also like to thank our teachers, friends, and families for their support, encouragement, and assistance throughout our academic journey.

Finally, we extend our thanks to everyone who contributed, directly or indirectly, to the realization of this project.

Dedication

I dedicate this work to my beloved parents, whose endless love, sacrifices, patience, and unwavering support have guided me throughout my academic journey. Their encouragement has always been my greatest source of motivation.

To my family, for their constant prayers, understanding, and confidence in my abilities.

To my friends and colleagues, for their support, advice, and the memorable moments we shared during these years of study.

Finally, to everyone who contributed, directly or indirectly, to the completion of this work, I express my sincere gratitude.

BEKKOUCHE

Dedication

I dedicate this modest work to my dear parents, whose love, dedication, and continuous encouragement have been the foundation of my success. I am deeply grateful for everything they have done for me.

To my brothers, sisters, and family members, for their support, patience, and belief in me throughout this journey.

To my friends and classmates, for their cooperation, motivation, and friendship that made this experience both enriching and enjoyable.

Finally, I would like to thank all those who contributed, in one way or another, to the realization of this work and to the achievement of this important milestone.

CHAREF

Abstract

Data quality has become a fundamental issue in modern information systems, due to the continuous growth of data generated from multiple heterogeneous sources. One of the most important tasks in this field is **Record Linkage (RL)**, which aims to identify records referring to the same real-world entity across one or several datasets. However, this process involves a high computational cost when applied to large volumes of data, as it requires performing comparisons between a very large number of records. To address this challenge, **blocking techniques** are commonly employed to reduce the search space by grouping potentially matching records into common blocks. The success of this operation depends largely on the appropriate selection of **blocking keys**, which has led to automatic blocking key selection being considered an important optimization problem. This dissertation proposes an integrated Record Linkage framework based on three complementary optimization techniques. First, the **Artificial Bee Colony (ABC) algorithm**, inspired by the foraging behavior of honeybee colonies, is used to automatically select the optimal subset of blocking keys that maximizes the **Pair Completeness (PC)** metric, using transformation functions such as Soundex, NYSIIS, First_N_Chars, and Last_N_Chars. Second, **K-Modes clustering** is applied to improve the balance of the generated blocks. Third, the **Woodpecker Optimization Algorithm (WPO)** is employed during the matching phase to optimize similarity weights and decision thresholds, thereby enhancing duplicate detection accuracy. The system was developed in **Python** and integrated into a user-friendly web application. The obtained results demonstrate the ability of this framework to identify effective blocking key subsets, reduce the search space, and improve the overall efficiency of the Record Linkage process while maintaining a high level of matching quality. These findings confirm the potential of **swarm intelligence** techniques and metaheuristic algorithms as promising solutions for data quality improvement in large-scale Record Linkage applications.

Keywords: Data Quality, Record Linkage, Deduplication, Blocking, Blocking Key Selection, Artificial Bee Colony (ABC), Woodpecker Optimization Algorithm (WPO), Swarm Intelligence, Pair Completeness (PC), Data Integration, Optimization.

Résumé

La qualité des données représente aujourd'hui un défi majeur dans les systèmes d'information modernes, en raison de l'augmentation rapide des données issues de sources multiples et hétérogènes. Parmi les principales techniques utilisées pour améliorer cette qualité, le **couplage d'enregistrements** (Record Linkage) permet d'identifier les enregistrements faisant référence à une même entité réelle. Cependant, ce processus devient coûteux en termes de calcul lorsqu'il est appliqué sur de grands volumes de données, car il nécessite un nombre important de comparaisons. Afin de résoudre cette problématique, les techniques de **Blocking** sont utilisées pour réduire l'espace de recherche en regroupant les enregistrements susceptibles de correspondre dans des blocs. La performance de cette étape dépend principalement du choix des **Blocking Keys**, ce qui transforme leur sélection en un problème d'optimisation. Dans ce mémoire, nous proposons un cadre intégré basé sur trois techniques d'optimisation. Premièrement, l'algorithme **Artificial Bee Colony (ABC)** est utilisé pour sélectionner automatiquement le meilleur sous-ensemble de clés de blocage en maximisant la métrique **Pair Completeness (PC)**, à travers des fonctions de transformation telles que **Soundex, NYSIIS, First_N_Chars et Last_N_Chars**. Deuxièmement, le clustering **K-Modes** est intégré afin d'améliorer l'équilibre des blocs générés et de réduire les comparaisons inutiles. Troisièmement, l'algorithme **Woodpecker Optimization Algorithm (WPO)** est appliqué dans la phase de **Matching** afin d'optimiser les poids des mesures de similarité et les seuils de décision, améliorant ainsi la précision de détection des doublons. Le système proposé a été développé en **Python** et intégré dans une application web facilitant son utilisation. Les résultats obtenus montrent que cette approche permet de réduire efficacement l'espace de recherche, d'améliorer les performances du processus de couplage d'enregistrements et de maintenir une bonne qualité de correspondance. Cette étude confirme ainsi le potentiel des techniques d'intelligence en essaim et des algorithmes métaheuristiques pour améliorer les systèmes de **Record Linkage** à grande échelle.

Mots-clés : Qualité des données, Couplage d'enregistrements, Déduplication, Blocage, Sélection des clés de blocage, Artificial Bee Colony (ABC), Algorithme d'optimisation du pic-bois (WPO), Intelligence en essaim, Pair Completeness (PC), Intégration des données, Optimisation.

الملخص

أصبحت جودة البيانات من المسائل الجوهرية في أنظمة المعلومات الحديثة، نظراً للتزايد المستمر في حجم البيانات المُنتَجة من مصادر متعددة ومتباينة. ومن أبرز المهام في هذا الإطار عملية **ربط السجلات** (Record Linkage)، التي تهدف إلى تحديد السجلات المتعلقة بالكيان الحقيقي ذاته عبر مجموعة بيانات واحدة أو أكثر. غير أن هذه العملية تنطوي على تكلفة حسابية مرتفعة حين تُطبَّق على مجموعات بيانات ضخمة، إذ يستلزم الأمر إجراء مقارنات بين أعداد كبيرة جداً من السجلات.

للتغلب على هذا التحدي، يُلجأ عادةً إلى تقنيات **التجميع الانتقائي** (Blocking)، التي تُسهم في تقليص فضاء البحث عن طريق تجميع السجلات المحتمل تطابقها ضمن كتل مشتركة. ويتوقف نجاح هذه العملية إلى حد بعيد على حسن اختيار **مفاتيح التجميع** (Blocking Keys)، وهو ما دفع إلى اعتبار الاختيار التلقائي لهذه المفاتيح مسألةً تحسينيةً بالغة الأهمية.

تقترح هذه المذكرة إطاراً متكاملاً لربط السجلات يعتمد على ثلاث تقنيات تحسينية مترابطة. أولاً، تُوظَّف **خوارزمية مستعمرة النحل الاصطناعية** (Artificial Bee Colony - ABC)، المستوحاة من سلوك النحل في البحث عن الغذاء، لاختيار المجموعة الفرعية المثلى من مفاتيح التجميع التي تُعظّم مقياس **اكتمال الأزواج** (Pair Completeness - PC)، وذلك باستخدام دوال تحويل من قبيل Soundex :و NYSIISو First_N_Charsو .Last_N_Charsثانياً، يُطبَّق **تجميع K-Modes** لتحسين توازن الكتل المُولَّدة. وثالثاً، يُستخدم **خوارزمية تحسين نقار الخشب** (Woodpecker Optimization Algorithm - WPO) في مرحلة المطابقة لضبط أوزان التشابه وعتبات القرار، مما يُعزز دقة اكتشاف التكرارات.

تم تطوير النظام بلغة **Python**وإدماجه ضمن تطبيق ويب سهل الاستخدام. وقد أثبتت النتائج المتحصَّل عليها قدرة هذا الإطار على تحديد مجموعات فعّالة من مفاتيح التجميع، وتقليص فضاء البحث، وتحسين كفاءة عملية ربط السجلات مع الحفاظ على مستوى عالٍ من جودة المطابقة.

تُكرّس هذه النتائج إمكانية تقنيات **ذكاء الأسراب** (Swarm Intelligence)والخوارزميات الميتاهيوريستية بوصفها حلولاً واعدة لتحسين جودة البيانات في تطبيقات ربط السجلات على النطاق الواسع.

الكلمات المفتاحية :جودة البيانات، ربط السجلات، إزالة التكرار، التجميع الانتقائي، اختيار مفاتيح التجميع، خوارزمية مستعمرة النحل الاصطناعية، خوارزمية تحسين نقار الخشب، ذكاء الأسراب، اكتمال الأزواج، تكامل البيانات، التحسين.

Table of Contents:

List of Figures:

list of tables:

General Introduction

In the era of digital transformation, organizations continuously generate and collect massive amounts of data from various sources. The integration of these heterogeneous data sources has become essential for supporting decision-making processes, knowledge discovery, and business intelligence. However, integrating data from multiple sources often introduces several data quality problems, including missing values, inconsistent formats, typographical errors, and duplicate records. Among these challenges, duplicate records represent one of the most significant issues affecting data quality. Duplicate records may refer to the same real-world entity but appear in different forms due to data entry errors, abbreviations, spelling variations, or inconsistencies between data sources. Such problems can negatively impact data analysis, reduce information reliability, and lead to incorrect decisions. To address this issue, Record Linkage (RL) has emerged as an important research area. Record Linkage aims to identify and link records that refer to the same real-world entity within a single database or across multiple data sources. It is widely used in various domains such as healthcare, e-commerce, census analysis, fraud detection, customer relationship management, and bibliographic databases. Despite its importance, Record Linkage remains a computationally expensive process, especially when dealing with large datasets. Performing comparisons between all possible record pairs leads to a significant increase in execution time and computational cost. Therefore, reducing the number of unnecessary comparisons while maintaining high matching accuracy has become a major challenge. To overcome this limitation, blocking techniques are commonly employed. Blocking partitions records into smaller groups called blocks, allowing comparisons to be performed only within the same block. The effectiveness of this phase strongly depends on the selection of appropriate blocking keys. Poor blocking key selection may either miss true matches or generate an excessive number of comparisons. In this dissertation, we propose a Record Linkage framework based on metaheuristic optimization techniques. First, the Artificial Bee Colony (ABC) algorithm is used to automatically select the most relevant blocking keys. Then, K-Modes clustering is applied to optimize the generated blocks and improve their balance. Finally, the Woodpecker Optimization Algorithm (WPO) is employed during the matching phase to optimize similarity weights and

decision thresholds, thereby improving matching accuracy and overall system performance. The proposed approach aims to improve the efficiency and effectiveness of the Record Linkage process by reducing unnecessary comparisons, preserving true matches, and enhancing duplicate detection accuracy.

This dissertation is organized into three chapters. The first chapter introduces the fundamental concepts related to data quality, Record Linkage, and blocking techniques. The second chapter presents the state of the art and reviews existing approaches in the literature. The third chapter describes the proposed methodology based on ABC, K-Modes, and WPO, along with its implementation and experimental results.

Chapter I: Fundamentals Concepts

1.1 Introduction

Data quality is a key element in any information management strategy, particularly in record linkage, which aims to connect data from different sources representing the same entity. However, this process may be affected by errors related to missing or inconsistent data, leading to inaccurate results and poor decision-making. Therefore, ensuring high data quality is essential for achieving reliable record linkage. This requires collaboration among the various stakeholders and the use of appropriate measures to assess and reduce errors. By applying these best practices, organizations can obtain more reliable and relevant data.

1.2 Data Heterogeneity

Here, we address a complex issue present in most data integration systems. This complexity arises from differences between systems, related to various stages, multiple viewpoints, and different design approaches adopted. It is therefore essential to study the forms of data heterogeneity that emerge during integration. Two main types of heterogeneity are generally distinguished :

1. Semantic Heterogeneity

This refers to situations where the same concept is expressed with different meanings [1]. Two types of semantic heterogeneity can be identified:

- **Schema-related semantic heterogeneity:** this occurs when the same concept is represented using different terminologies.
- **Data-related semantic heterogeneity:** this occurs when data coming from different sources have different origins, are stored in different containers, and follow different structures and conventions [2].

2. Structural Heterogeneity

Structural heterogeneity, also referred to as schema heterogeneity, describes a situation where the same concepts can be represented in different ways. This results in the use of various models to describe the same data, or conversely, in different representations for identical content.

There are four main types of semantic heterogeneity (semantic conflicts) [3]:

- **Representation conflicts:** occur when different schemas or different attributes are used to describe the same concept.
- **Naming (term) conflicts:** these arise when different names are used for the same concept or attribute (synonyms), or when identical names refer to different concepts (homonyms).
- **Context conflicts:** these occur when different representations are provided for a single object across data sources, where each source has its own local context for that object.
- **Value measurement conflicts:** these arise when different units are used to measure values of the same concepts.

1.3 Big Data

In recent years, the definition of the term "Big Data" has generated differing opinions among researchers and companies. Some emphasize the massive volume of data, while others focus more on the technologies used to analyze it and extract value. The earliest definitions were proposed by the Gartner Group in 2001, with the "3Vs" model: Volume, Velocity, and Variety. This definition was later enriched in 2012 and further extended by IBM with the addition of a fourth V: Veracity.

1.3.1 Volume

The term "volume" refers to the enormous amount of data collected from various sources. The purpose of gathering and analyzing these large datasets is to extract as much knowledge as possible for researchers and organizations [4].

1.3.2 Variety

The term "variety" refers to the diversity of data types collected. Big Data analysis relies on data coming from multiple sources, which may be structured (such as relational databases), semi-structured, or unstructured, such as text files.

1.3.3 Velocity

This refers to the speed at which data is generated and transmitted to an organization, as well as the time required to process and analyze it.

1.3.4 Veracity

Veracity refers to the credibility and relevance of data for the target audience. Organizations collect Big Data that is not always directly supervised by specialists; therefore, it may contain errors. Data credibility is thus a crucial factor in drawing accurate conclusions. It is also noted that veracity includes considerations related to data privacy and legal concerns [5].

1.4 Definition of Data Quality

Data quality refers to the measure of the condition of data based on various factors such as accuracy, completeness, consistency, reliability, and timeliness. Assessing data quality levels helps organizations identify potential errors that need to be corrected and determine whether the data within their information systems is sufficient for their needs. In other words, data quality is the evaluation of the reliability and relevance of the information contained in a dataset [6].

Data quality is an important concern for both businesses and governments, as it directly affects decision-making and operational efficiency. Poor-quality data can lead to errors, inefficiencies, and costs for organizations. Data quality management involves the development of processes and methods to ensure that data is of the highest possible quality, as well as the implementation of measures to monitor and maintain data quality over time [7].

1.5 Data Quality Criteria

1.5.1 Intrinsic Criteria

- **Completeness:** Completeness refers to the extent to which data contains all the information required for its use. Complete data has no missing values and provides a comprehensive and sufficient view of the object or phenomenon under study, enabling

informed decision-making [8].

- **Validity:** Validity corresponds to compliance with predefined rules, formats, or constraints. It is measured by the extent to which data adheres to these requirements, thereby ensuring its conformity and reliability.

- **Accuracy:** Accuracy expresses the degree to which data faithfully represents the reality it describes. Accurate data is free from errors, bias, or distortion, ensuring the relevance of analyses and decisions.

- **Uniqueness:** Uniqueness refers to the absence of duplicates or redundancies in data. Each piece of information must be unique and identifiable in order to avoid ambiguity or confusion during its use.

- **Consistency:** Consistency concerns the harmonization of data according to common rules and formats. Consistent data does not present contradictions and facilitates integration, comparison, and analysis.

- **Timeliness:** Timeliness refers to how recent and up-to-date the data is. Up-to-date data accurately reflects the current situation and enables fast, relevant, and effective decision-making.

1.5.2 Service Criteria

- **Relevance:** Relevance measures the actual usefulness of data for users and business processes. Data is considered high quality if it is appropriate for its intended use and provides real value, without being unnecessarily detailed or redundant.

- **Understandability:** Understandability reflects users' ability to comprehend the data. To achieve this, terms, concepts, and attributes must be clearly defined and aligned among all stakeholders. The use of business glossaries, data dictionaries, and usage inventories

facilitates this shared understanding.

- **Accessibility:** Accessibility refers to the ease with which data can be accessed and used. It involves knowing the exact location of the information and its source of truth within the information system. Depending on requirements, access may be provided in event mode (at each update), query mode (on demand), or batch mode for large-scale synchronization.

1.6 The Importance of Data Quality

Poor-quality data can have significant consequences for an organization, ranging from operational disorder to incorrect business decisions and inaccurate analyses. Data quality issues can also generate additional costs, such as shipping products to wrong addresses, losing business opportunities due to incomplete or incorrect contact information for prospects, or facing penalties for inaccurate financial or regulatory reporting [10].

We also consider that distrust among executives and managers regarding data quality is one of the main barriers to the adoption of Business Intelligence and analytical tools as decision-support systems within organizations.

1.7 Data Quality Issues

Data quality problems do not occur randomly; their causes are well identified. They may be of technical or human origin. These malfunctions gradually accumulate throughout the entire data lifecycle, from data creation, through data manipulation, to data usage and analysis [11].

1.7.1 Data Creation:

Data creation involves the design and modeling of the database up to data entry by users. During this stage, several sources of problems have been identified:

1) Manual input: lack of systematic validation of input forms.
2) Automatic input: OCR capture issues, speech recognition errors, incompleteness, lack of normalization, inadequate conceptual data modeling, poorly structured attributes, and absence of integrity constraints to maintain data consistency.

3) Duplicate entries.
4) Approximations.
5) Hardware or software limitations.
6) Measurement errors.
7) Data corruption: physical or logical security breaches affecting data integrity.

1.7.2 Data Collection and Import:

1) Destruction or loss of information due to inappropriate preprocessing.
2) Data loss: buffer overflows and transmission problems.
3) Lack of verification in bulk import procedures.
4) Introduction of errors by data conversion programs.

1.7.3 Data Storage:

1) Absence of metadata.
2) Lack of updating and refreshing of obsolete or replicated data.
3) Ad-hoc modifications.
4) Inappropriate data models and structures, incomplete specifications, or evolving analytical and system design requirements.
5) Hardware or software constraints.

1.7.4 Data Integration:

1) Integration problems involving multiple data sources with different quality and aggregation levels.
2) Temporal synchronization issues.
3) Non-standard data systems.
4) Sociological factors leading to interpretation and integration issues.

1.7.5 Data Search and Analysis:

1) Human error.
2) Constraints related to computational complexity.

3) Software limitations and incompatibility issues.
4) Scalability and performance issues, as well as lack of trust in results.
5) Approximations due to dimensionality reduction techniques.

1.8 General Approaches for Detecting and Correcting Data Quality Problems

As illustrated in Figure 1.1, most research work addressing the issue of data quality can be grouped into four main complementary approaches [15].

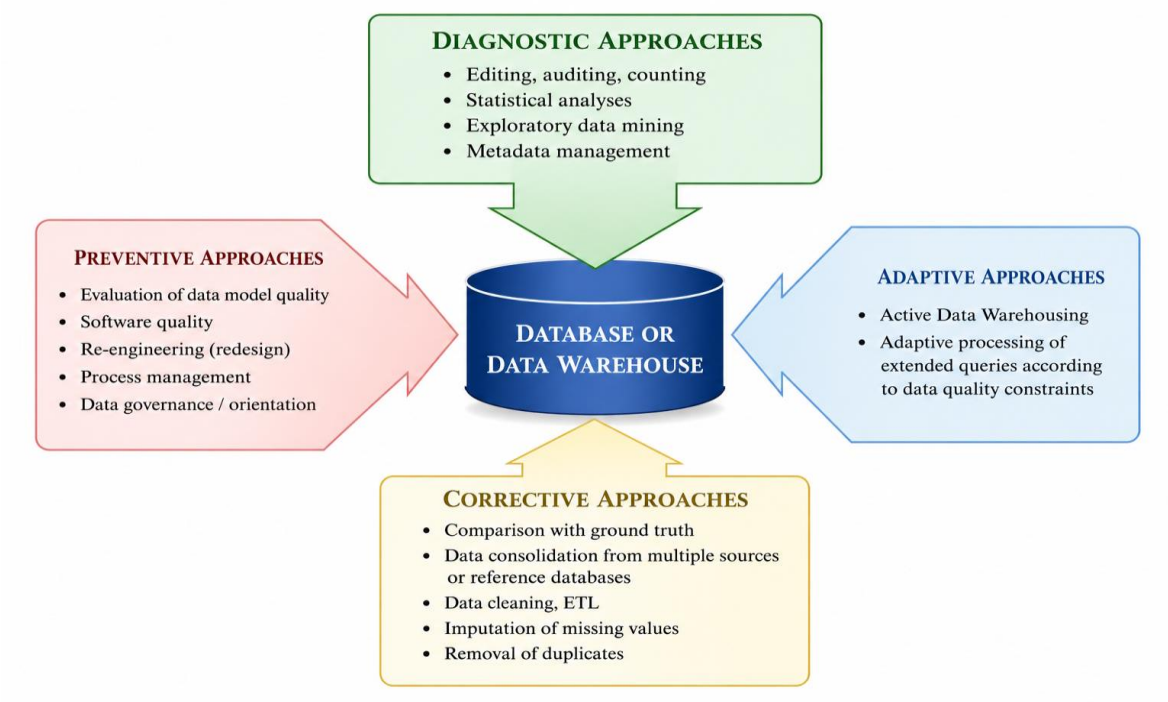

Figure 1.1: Overview of Approaches for Data Quality Assessment and Control

- **Preventive approaches:** focused on information systems engineering and process control, using techniques that evaluate the quality of conceptual models, software development, and the processes employed for data processing.
- **Diagnostic approaches:** centered on statistical, analytical, and exploratory data mining methods used to detect anomalies in data.

- **Corrective approaches:** focused on data cleaning and consolidation techniques, using extended data manipulation languages and ETL tools (Extraction–Transformation–Loading).

1.9 Data Integration

Data integration refers to a set of processes that allow data from different sources to be grouped and merged within a unified interface. It involves eliminating conflicts and inconsistencies in order to present data in a homogeneous way. Thus, users can access information through a global view, which facilitates query processing and data analysis [12], [13].

A data integration system is composed of three main layers:

1) **Global schema layer:** it may consist of a data warehouse (materialized approach) or a mediator (virtual approach). This layer allows users to query data by accessing sources through a global schema.
2) **Adapters/loaders or wrappers layer:** the adapter extracts data through the global schema and enables interaction with the data sources according to the upper layers. The adapter is the only means of accessing data sources and retrieving information.
3) **Data sources layer:** it consists of the selected data sources to be integrated.

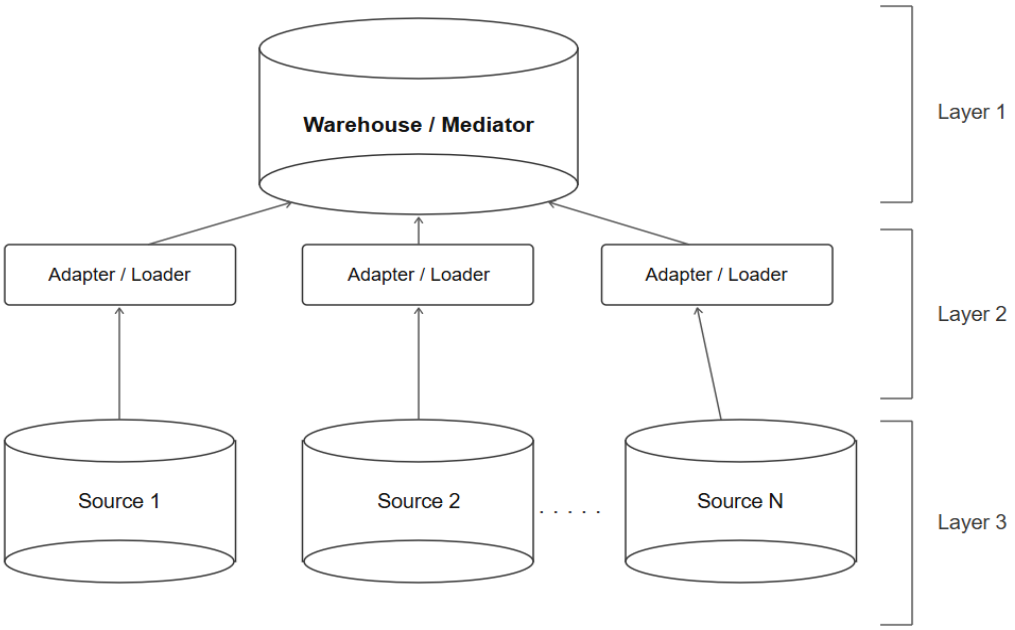

Figure 1.2: General Architecture of a Data Integration System

Alternatively, a data integration system can be defined by the triple (G, S, M):

G: represents the global schema.

S: represents the set of data sources.

M: refers to the mapping between the global schema and the set of data sources.

1.10 Data Deduplication

Data deduplication is a complex process aimed at reducing redundancy in order to improve backup management and optimize storage space. Widely used in many backup solutions, this technique attracts users who seek to compare the performance of different products through deduplication rates. However, these rates may vary depending on the measurement methods used and several factors influencing their calculation, which makes them sometimes unreliable.

Therefore, when comparing solutions, it is preferable to focus on the actual storage space used after deduplication rather than relying solely on the reported rates [14].

1.11 Conclusion

In this chapter, we have highlighted the critical importance of data quality in information systems, particularly in the context of data integration and record linkage. We have shown that data, often originating from heterogeneous sources, may present problems related to its structure, semantics, or representation, which complicate its use.

Furthermore, we have emphasized that data quality relies on several essential criteria such as accuracy, completeness, consistency, and relevance and that poor quality can have significant consequences on organizational performance and decision-making. The causes of these issues, whether technical or human, occur throughout the entire data lifecycle.

We have also presented the main approaches for detecting and correcting these problems, as well as key concepts related to data integration and deduplication, which play a central role in improving overall data quality. Thus, ensuring reliable, consistent, and usable data represents a major challenge for organizations and an essential step toward meaningful analysis and effective decision-mak

Chapter II: Record Linkage

2.1 Introduction

Record Linkage (RL) is an essential process in the field of data quality. Its main objective is to identify tuples corresponding to the same real-world entity and then merge them into a single record. This technique significantly improves data quality by eliminating duplicates and ensuring a more consistent representation of information.

2.2 Definition

Record Linkage refers to the process of identifying, within one or more datasets, the records that correspond to the same real-world entity, even if they originate from different sources. This operation is essential in data integration, particularly when records do not share a common identifier due to differences in formats, storage methods, or conventions used. In computer science, this technique is also known as data matching or deduplication when it is used to detect duplicates within the same source [16].

Thus, the main objective of Record Linkage is to identify tuples corresponding to the same real-world entity in order to group or merge them. This process significantly improves data quality by eliminating redundancies and ensuring a more reliable and consistent representation of information.

2.3 Types of Record Linkage

Two main types of Record Linkage (RL) can be distinguished: exact matching and statistical matching. Exact matching itself includes two subcategories: deterministic linkage and probabilistic linkage, as illustrated in the figure below [17].

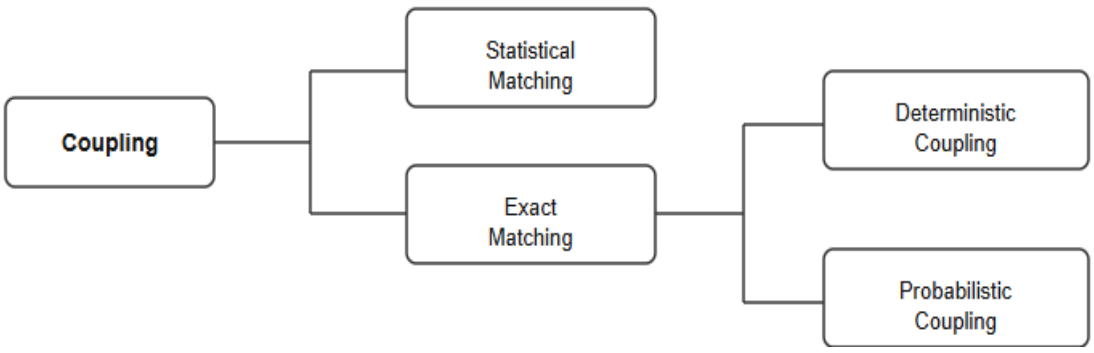

Figure 2.1: Types of Record Linkage

2.3.1 Statistical Matching

Statistical matching aims to build a dataset that accurately represents the overall structure of a population. In this case, the linked records do not necessarily correspond to the same real-world entity, such as a person or a company, but may come from different units while still referring to the same population. This approach is based on the assumption that the relationships between observed variables in the population are similar to those found in the available datasets. It is mainly used in market research and is less commonly employed by official statistical agencies.

2.3.2 Exact Matching

The objective of exact matching is to link information from a record in one dataset with that of another dataset in order to build a reliable and consistent dataset for each record. This process is performed at the individual level, for example by linking mortality records with census records.

There are two main methods of exact matching:

2.3.2.1 Deterministic Matching

This is the simplest form of Record Linkage, based on the use of identifiers or common variables across different data sources. However, it is rare to have a single, reliable, and sufficiently discriminative variable to correctly identify records. In most cases, a combination of several variables is required to distinguish entities.

This method is widely used by official statistical agencies. For instance, Statistics Canada relies on this approach to build its business, address, and population registers, which subsequently requires several validation and survey operations [18].

2.3.2.2 Probabilistic Matching

This is another form of exact matching. As in the deterministic case, no unique identifier is available to perform the linkage. However, unlike deterministic matching, probabilistic matching can handle incomplete or error-prone data.

Thus, records that do not perfectly match across all variables can still be linked as potential pairs. For each pair, a similarity score is computed, and a linkage decision is then made based on this score [18].

2.4 Record Linkage Steps

Record Linkage can be defined as a three-step process [19], as illustrated in **Figure 2.2.**

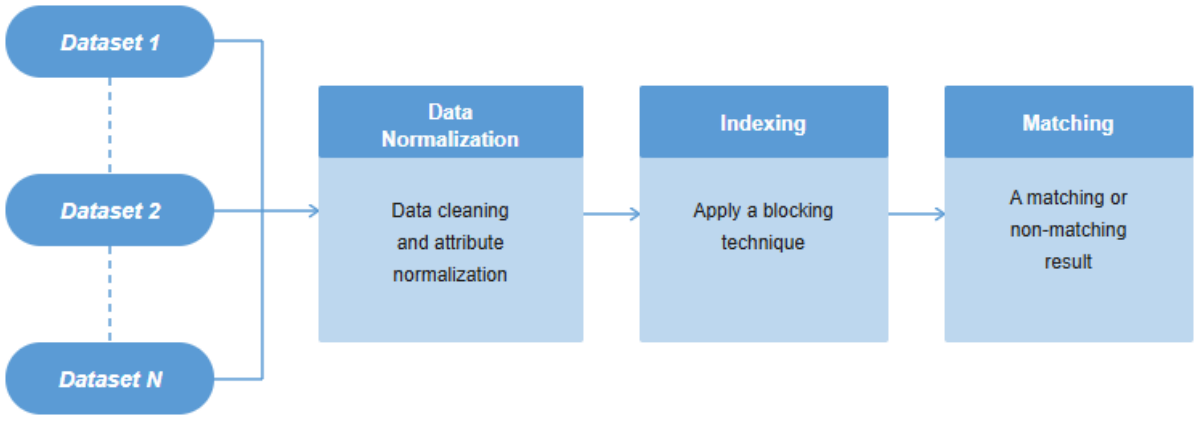

Figure 2.2: Steps of Record Linkage

2.4.1 Cleaning and Normalization

Applying the Record Linkage (RL) process on poor-quality data may lead to errors such as incorrect merging of tuples and loss of important information. For example, the attribute “address” may be stored as a single field in one database, while it is split into multiple fields (postal code, street, city, etc.) in another.

Therefore, in order to ensure the effectiveness of the linkage process, it is necessary to perform data normalization, particularly for the address field, before starting the RL process [20].

2.4.2 Indexing

This step is considered the most important in the process. It aims to identify the most representative elements of a document's content. Among the most commonly used indexing techniques is **blocking**, which helps to efficiently organize data [21].

2.4.2.1 Blocking

Blocking is the most widely used method during the indexing stage. It consists of partitioning the dataset into several blocks so that tuples belonging to the same block share a common value called the **Blocking Key Value (BKV)**. This key can be defined using a single attribute [22].

The table illustrates an example of blocking keys generated from a restaurant dataset, where the blocking key is constructed by concatenating the restaurant's city with its phone number.

BK1	BK2	Name	Address	City	Phone	Type
A6553102461501	LASANG435	arnie morton's of Chicago	435 s. la cienega blv.	Los Angeles	310/246-1501	American
H3413104721211	STADAC12224	art's deli	12224 ventura bold	Studio city	818-762-1221	delis

Table 2.1: Example of Blocking Key

2.4.2.1.1 Phonetic Encoding

A - Soundex

The main steps of the Soundex algorithm are:

- Keep the first letter of the string unchanged.

- Replace all consonants using the following rules: (0 for A, E, H, I, O, U, W, Y; 1 for B, F, P, V; 2 for C, G, J, K, Q, S, X, Z; 3 for D, T; 4 for L; 5 for M, N; 6 for R).
- If the resulting string is too short, the algorithm fills the remaining positions (after the first character) with zeros [23].

B - NYSIIS (New York State Identification and Intelligence System)

The basic rules of the NYSIIS algorithm involve transforming initial and final characters. For example, initial transformations include: MAC is replaced by MCC, KN becomes NN, K is replaced by C, PH is transformed into PF, and SCH becomes SSS. Final character transformations include: EE and IE are replaced by Y, while DT, RT, RD, NT, and ND are all replaced by D [23].

2.4.3 Matching

The third and final step of the Record Linkage process consists of comparing indexed records within the same block in order to determine whether they correspond to the same real-world entity. The similarity score is usually normalized between 0 and 1, where 1 indicates a perfect match and 0 indicates no similarity at all.

This comparison can be performed using different string similarity functions, such as the Levenshtein distance (1966), or by using machine-learning algorithms to classify pairs of records as matches or non-matches.

2.4.3.1 Pattern Matching

A - Edit Distance

The edit distance, also known as the Levenshtein distance, was introduced in 1965 by Vladimir Levenshtein. It is one of the most commonly used measures for evaluating the similarity between two strings. It corresponds to the minimum number of operations—insertions, deletions, or substitutions—required to transform one string into another. To better illustrate this concept, the following

example shows the computation of the costs needed to transform one word into another.

Word 1	Word 2	Operation	Cost
I		Delete (I)	1
N	E	Substitute (E)	1
T	X	Substitute (X)	1
E	E	Comparison	0
	C	Insert (C)	1
N	U	Substitute (U)	1
T	T	Comparison	0
I	I	Comparison	0
O	O	Comparison	0
N	N	Comparison	0
		Sum	5

Table 2.2: Cost calculation for transforming one word into another

Thus, the edit distance between the two words “Intention” and “Execution” corresponds to the minimum cost required to transform the first string into the second, namely a total of five operations. The steps presented in the example do not represent the only possible solution, but they illustrate the transformation with the lowest cost.

B - Jaro-Winkler

Jaro-Winkler is a string similarity metric proposed by William E. Winkler in 1990 as an extension of the Jaro distance. To compute the Jaro-Winkler similarity

between two strings, the first step is to calculate the traditional Jaro similarity, which is defined as follows:

$$Jaro - Sim(s_1, s_2) = \begin{cases} 0, & m = 0 \\ \frac{1}{3}\left(\frac{m}{s_1} + \frac{m}{s_2} + \frac{m-t}{m}\right) & otherwise \end{cases}$$

- **s**: represents the length of the string.
- **m**: represents the number of common characters between the compared sequences with the same index.
- **t**: represents half the number of transpositions.

To improve the previous metric, William E. Winkler introduced a prefix scale **P** in order to give higher similarity scores to strings that share the same prefix **L**, up to a maximum length of four characters. The Jaro-Winkler similarity is defined as follows [24]:

$$JaroWinlker - Sim(s_1, s_2) = Jaro - Sim(s_1, s_2) + LP(1 - Jaro - Sim(s_1, s_2))$$

Where:

- **Jaro-Sim (s1, s2)** is the Jaro similarity between the two strings.
- **L** is the prefix length.
- **P** is a scaling factor (a constant, typically set to 0.1).

C - Jaccard Distance

The Jaccard distance is generally used to measure the similarity between two sample sets, which can also include strings. To compute the Jaccard distance, the Jaccard coefficient must first be calculated, which is defined as follows [25]:

$$Jaccard(A, B) = \frac{A \cap B}{A \cup B}$$

Once this is done, the Jaccard distance is obtained by subtracting the Jaccard coefficient from 1.

$$Jaccard_{distance}(A, B) = 1 - Jaccard(A, B)$$

2.5 Conclusion

This chapter presented the main stages of the Record Linkage process, with a particular focus on data normalization, blocking-based indexing, and record comparison. It also highlighted the importance of handling heterogeneous data and the use of similarity measures such as the Levenshtein distance.

In summary, these concepts provide an essential foundation for ensuring reliable and effective record linkage.

Chapter III: Design and Implementation

3.1 Introduction

In this chapter, we present the development environment and the technologies used for the implementation of the proposed Record Linkage framework. We then describe the proposed methodology, including the Artificial Bee Colony (ABC) algorithm for automatic blocking key selection, K-Modes clustering for block optimization, and the Woodpecker Optimization Algorithm (WPO) for matching optimization. Finally, we present and discuss the experimental results, followed by screenshots illustrating the main functionalities of the developed application.

3.2 Automatic Blocking Key Selection – The ABC Algorithm

3.2.1 Definition of the ABC Algorithm

The Artificial Bee Colony (ABC) algorithm is a swarm-based meta-heuristic optimization algorithm introduced by Karaboğa [2] at Erciyes University. It belongs to the family of Swarm Intelligence (SI) algorithms, which solve complex optimization problems by simulating the collective behavior of social organisms [29].

In the ABC model, the colony consists of three groups of artificial bees: employed bees, onlooker bees, and scout bees. It is assumed that there is exactly one employed bee per food source, meaning the number of employed bees equals the number of candidate solutions in the population. The first half of the swarm consists of employed bees, and the second half constitutes the onlooker bees [28].

3.2.2 Biological Inspiration

The ABC algorithm is specifically inspired by the foraging behavior of honeybee colonies, based on the model proposed by Tereshko and Loengarov (2005). In nature, when scout bees discover a food source, they return to the hive and perform a "waggle dance" to communicate its location and quality to other bees. The better the food source, the longer and more energetic the dance, attracting more onlooker bees to exploit it.

This process is driven by two self-organizing feedback mechanisms:

- **Positive feedback**: Recruitment of bees toward rich food sources.
- **Negative feedback**: Abandonment of exhausted or poor food sources.

The three types of bees and their roles are as follows:

Bee Type	Role in the Algorithm
Employed Bee	Exploits a known food source and shares its quality with onlookers
Onlooker Bee	Observes employed bee dances and selects sources proportional to their quality
Scout Bee	Replaces abandoned food sources with new randomly discovered ones

The neighborhood search performed by each employed bee is defined by the following perturbation formula proposed in the ABC algorithm [29]:

$$v_{ij} = x_{ij} + \phi_{ij} \left(x_{ij} - x_{kj} \right)$$

Where:

- **xij** is the current solution (food source)
- **xkj** is a randomly chosen distinct solution ($k \neq i$)
- **j** is a randomly selected dimension
- **Φij** is a random number in $[-1, 1]$

If the new solution ***vij*** is better than ***xij***, it replaces it (greedy selection). Otherwise, the trial counter of ***xij*** is incremented. When the trial counter exceeds a predefined threshold called ***limit***, the food source is abandoned and replaced by a new random solution generated by a scout bee.

3.2.3 Application to the Record Linkage Problem

The blocking key selection problem in Record Linkage can be naturally modeled as a feature selection problem, since selecting the best subset of blocking keys from a larger set of candidates is equivalent to finding the most informative feature subset. This makes it directly compatible with the ABC algorithm framework, which has been successfully applied to feature selection tasks in various domains [26], [30].

In our approach, the ABC algorithm is adapted to the blocking key selection problem as follows:

- **Food source:** Each candidate solution represents a subset of blocking keys selected from a pre-generated list of candidates (produced using transformation functions such as Soundex, NYSIIS, First_N_Chars, Last_N_Chars).

  - **Nectar amount (fitness):** The quality of each blocking key subset is evaluated using a fitness function based on both **Pair Completeness (PC)** and **Reduction Ratio (RR)**. Pair Completeness measures the proportion of true duplicate pairs preserved within the generated blocks, while Reduction Ratio measures the reduction in the number of record comparisons.

- **Neighborhood search:** A new candidate subset is generated by replacing one blocking key index within the current solution using the ABC perturbation formula, then rounding and clipping the result to valid key indices.
- **Scout phase:** If a blocking key subset fails to improve after ***limit*** consecutive trials, it is discarded and replaced by a new randomly generated subset.

The general flow of our proposed approach is:

1. Pre-process the dataset: remove attributes with low completeness or very low cardinality.
2. Generate the full list of candidate blocking keys using the defined transformation functions.
3. Initialize the ABC population as random subsets drawn from the candidate key list.
4. Run the ABC algorithm for *T* iterations, using Pair Completeness (PC) as the fitness function.
5. Return the best blocking key subset found across all iterations.

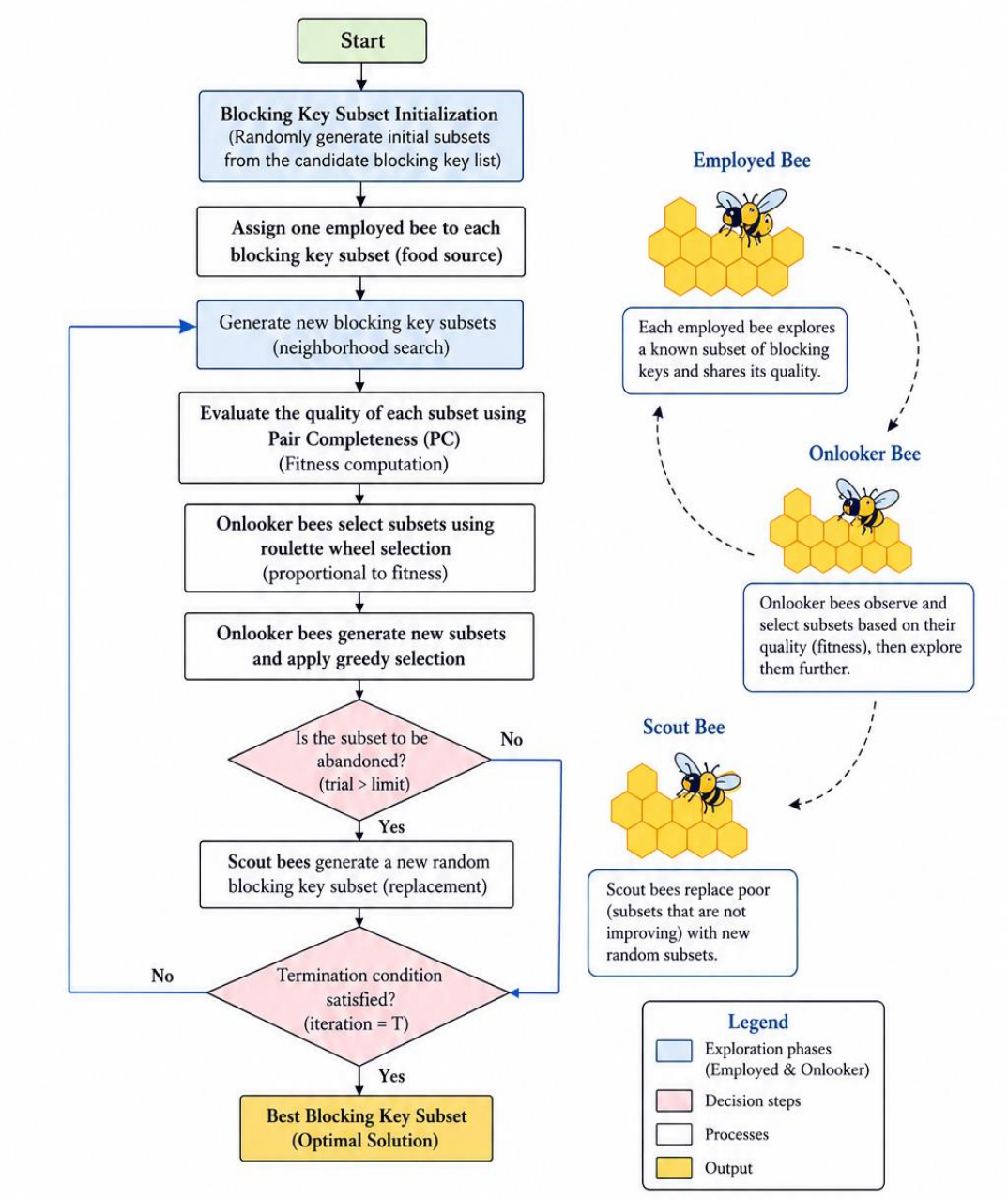

Figure 3.1: General Workflow of the Artificial Bee Colony (ABC) Algorithm for Blocking Key Selection

3.2.4 Evaluation Metrics for Blocking Key Selection

To evaluate the quality of the blocking key subsets generated by the ABC algorithm, three performance metrics were considered:

- **Pair Completeness (PC)**

Pair Completeness measures the ability of the blocking strategy to place true duplicate pairs within the same block. It is defined as:

PC = (Number of true duplicate pairs placed in the same block) / (Total number of true duplicate pairs)

A high PC value indicates that most duplicate records are preserved after the blocking phase.

- **Reduction Ratio (RR)**

Reduction Ratio measures the effectiveness of blocking in reducing the number of record comparisons. It is defined as:

RR = 1 − (Number of candidate comparisons after blocking / Total possible comparisons)

A higher RR value indicates a greater reduction in computational cost.

- **Fitness Function**

To evaluate each candidate blocking key subset, the ABC algorithm uses a fitness function based on both Pair Completeness (PC) and Reduction Ratio (RR).

The fitness value is computed as follows:

Fitness = (2 × PC × RR) / (PC + RR)

This harmonic mean ensures a balance between duplicate preservation and comparison reduction. A high fitness value can only be achieved when both PC and RR are simultaneously high.

3.2.5 K-Modes Clustering for Block Optimization

During the experimental phase, it was observed that the blocking keys generated by the ABC algorithm produced a large number of blocks with highly heterogeneous sizes. Some blocks contained only a few records, while others contained a very large number of records. This imbalance negatively affected the efficiency of the Record Linkage process and increased the computational cost of record comparisons.

To address this issue, the K-Modes clustering algorithm was incorporated into the proposed approach. Unlike K-Means, which is designed for numerical data, K-Modes is specifically adapted to categorical datasets and therefore fits the characteristics of Record Linkage data [33].

The selected blocking key attributes are used as input features for the K-Modes algorithm. The algorithm partitions the records into K clusters by grouping together records with similar categorical values. As a result, the generated clusters act as more balanced blocks, reducing the disparity in block sizes [33, 34] and improving the distribution of records across blocks.

The integration of K-Modes offers several advantages:

- Reduction of excessively large blocks.
- Better balance between block sizes.
- Reduction of unnecessary record comparisons.
- Improvement of the overall efficiency of the Record Linkage process.

The overall workflow of the proposed approach is therefore composed of two main stages. First, the Artificial Bee Colony (ABC) algorithm identifies the optimal subset of blocking keys. Second, K-Modes clustering is applied using the selected blocking key attributes in order to generate balanced blocks before the matching phase.

3.2.6 ABC Pseudo Code

For the implementation of the Artificial Bee Colony (ABC) algorithm applied to the automatic blocking key selection problem, each solution represents a subset of blocking keys selected from the list of candidate keys generated previously. The quality of each solution is evaluated using the Pair Completeness (PC) metric. The general pseudo-code of the ABC algorithm adapted to this problem is presented in the following algorithm.

```
Algorithm 1. ABC for Blocking Key Selection

Input:
    Dataset D, Candidate Blocking Keys (CBK),
    Population Size SN, Iterations T, Limit

Output:
    Best Blocking Key Subset

Initialize population
Evaluate fitness (PC)
Store best solution

While iteration ≤ T do

    Employed Bee Phase
        Generate neighboring solutions
        Apply greedy selection

    Onlooker Bee Phase
        Select solutions based on probabilities
        Generate neighboring solutions
        Apply greedy selection

    Scout Bee Phase
        Replace abandoned solutions

    Update best solution

End While

Return best solution
```

Algorithm 1. Pseudo-code of the Artificial Bee Colony (ABC) algorithm for automatic blocking key selection.

The first step consists of generating an initial population composed of blocking key subsets randomly selected from the candidate blocking key list generated previously. Each subset represents a food source in the Artificial Bee Colony (ABC) algorithm. The quality of each solution is evaluated using two blocking performance metrics: Pair Completeness (PC) and Reduction Ratio (RR). Pair Completeness measures the ability of the selected blocking keys to place true duplicate pairs within the same block, while Reduction Ratio measures the reduction in the number of record comparisons achieved by the blocking process.

The next stage corresponds to the **Employed Bee Phase**. For each food source, a new solution ***Pnew*** is generated using Equation (1). This equation explores the neighborhood of the current solution by modifying one of the blocking key indices. The parameter ***Φij*** is a random number in the interval ([-1, 1]), while ***Xkj*** represents a randomly selected neighboring solution. Once the new solution is generated, its fitness value is evaluated and compared with that of the current solution. If it performs better, it replaces the current solution according to the greedy selection principle.

During the **Onlooker Bee Phase**, each solution is assigned a selection probability proportional to its fitness value. Consequently, higher-quality solutions have a greater chance of being selected. The onlooker bees then generate new neighboring solutions from the selected food sources and apply greedy selection to retain the best solutions.

The final stage corresponds to the **Scout Bee Phase**. When a solution fails to improve after a predefined number of trials (***limit***), it is considered abandoned. It is then replaced by a new randomly generated blocking key subset. This mechanism helps maintain population diversity and prevents premature convergence to local optima.

The algorithm is executed until the maximum number of iterations is reached. At each iteration, the best solution is updated. At the end of the optimization process, the blocking key subset with the highest Pair Completeness value is returned as the optimal solution.

$$v_{ij} = x_{ij} + \phi_{ij}\left(x_{ij} - x_{kj}\right)$$

Overall, the workflow of the Artificial Bee Colony (ABC) algorithm for automatic blocking key selection is summarized in the flowchart shown in **Figure 3.**

3.3 Woodpecker Optimization Algorithm for Matching Optimization

3.3.1 Definition

The **Woodpecker Optimization Algorithm (WPO)** is a nature-inspired metaheuristic optimization algorithm that imitates the natural behaviors of woodpeckers. Similar to other population-based optimization algorithms, WPO maintains a set of candidate solutions and iteratively improves them according to a fitness function in order to find an optimal solution. Nature-inspired optimization algorithms have been widely used to solve complex optimization problems by transforming biological behaviors into mathematical search mechanisms [35] [36].

In the proposed Record Linkage framework, WPO is applied during the **matching phase** to optimize the matching parameters. Each woodpecker represents a candidate solution containing the weights assigned to similarity measures and the decision threshold used to classify record pairs as matches or non-matches.

3.3.2 Biological Inspiration

The inspiration behind WPO comes from the natural behavior of woodpeckers, especially their drumming and searching behaviors. Woodpeckers use repeated pecking sounds as a communication mechanism to attract mates, establish territories, and interact with their environment [37]. These behaviors provide a natural model for designing optimization strategies based on exploration and exploitation.

In WPO, the movement and interaction of woodpeckers are transformed into a search process. Candidate solutions explore different regions of the search space, and the algorithm gradually improves these solutions by selecting the most promising ones according to their fitness values [35].

3.3.3 Application to Matching

The matching phase of Record Linkage requires combining several string similarity measures into a single global similarity score in order to determine whether a given pair of records refers to the same real-world entity. Instead of manually fixing one similarity measure for each attribute, the proposed approach allows every combination of **(blocking attribute, similarity measure)** to compete independently. The WPO algorithm then learns which combination contributes the most to matching quality.

Let:

$$key\_cols = \{c_1, c_2, ..., c_k\}$$

Denote the set of attributes selected by the ABC blocking phase, and let:

$$M = \{JW, LEV, JAC\}$$

Represent the three considered similarity measures: **Jaro-Winkler, Levenshtein, and Jaccard**.

For every attribute ***cj*** and every measure $m \in M$, an independent similarity feature is computed for each candidate record pair ***(ra, rb)***:

$$S_{c_j,m}(r_a, r_b) = m(r_a[c_j], r_b[c_j])$$

This produces a feature vector with dimension: ***d = k x 3***

For each candidate pair, where ***(k)*** represents the number of blocking attributes.

The global similarity score is then calculated as a weighted sum over all **(attribute, measure)** combinations:

$$Sim(r_a, r_b) = \sum_{j=1}^{k} \sum_{m \in M} w_{j,m} \cdot S_{c_j,m}(r_a, r_b)$$

Subject to the normalization constraint:

$$\sum_{j,m} w_{j,m} = 1, \quad w_{j,m} \in [0,1]$$

Each woodpecker therefore represents a candidate solution vector with dimension ***(d+1)***, composed of the (d) similarity weights and the decision threshold ***(T)***:

$$X_{bird} = (w_{1,JW}, w_{1,LEV}, w_{1,JAC}, ..., w_{k,JW}, w_{k,LEV}, w_{k,JAC}, T)$$

After each positional update, a repair operator projects the candidate solution back into the feasible region. The weight components are limited to the range ([0, 1]) and normalized so that their sum equals one, while the threshold component is restricted to the predefined search interval: ***[Tmin, Tmax]***

This ensures that every evaluated solution remains a valid and normalized configuration throughout the optimization process.

A candidate pair is classified as a match when the global similarity score reaches the decision threshold:

$$Match(r_a, r_b) = 1 \quad if \quad Sim(r_a, r_b) \geq T$$
$$Match(r_a, r_b) = 0 \quad otherwise$$

The fitness value of each woodpecker is defined using the **F1-Score** obtained by applying its weight vector and threshold to all candidate pairs generated during the blocking phase and comparing the results with the ground truth:

$$Fitness = F1\text{-}Score = \frac{2 \times Precision \times Recall}{Precision + Recall}$$

Once the optimization process is completed, the learned weight vector does not need to keep all (***k x 3***) components for interpretation. For each attribute (***cj***), only the similarity measure with the highest learned weight is retained. The corresponding weights are then renormalized so that their sum equals one.

3.3.4 Workflow

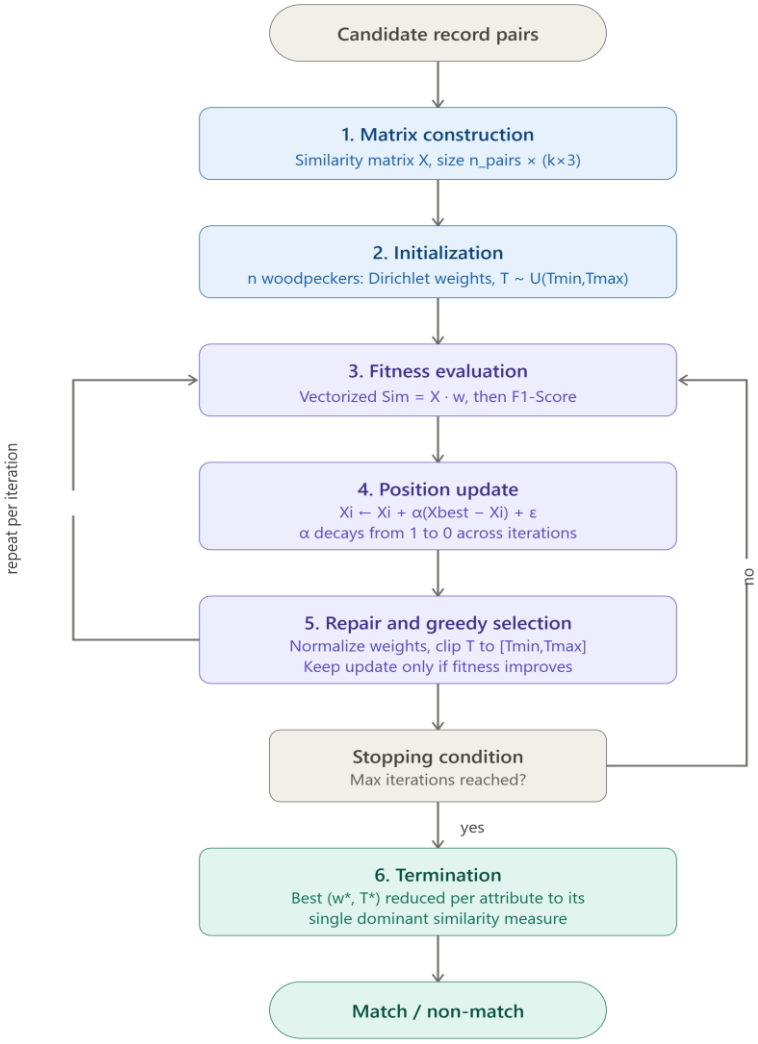

Figure 3.2: Workflow of WPO-based Record Linkage Optimization

The optimization procedure proceeds through the following stages:

1. Matrix Construction: For every candidate pair generated by the blocking phase, the similarity feature matrix ***X*** (size ***n_pairs*** × ***d***, where ***d = k × 3***) is computed across all (attribute, measure) combinations.

2. Initialization: A population of n woodpeckers is generated. For each one, the d weights are drawn from a Dirichlet distribution (ensuring they are non-negative and sum to one), and the threshold ***T*** is drawn uniformly from [***Tmin, Tmax***].

3. Fitness Evaluation: The fitness (***F1-Score***) of each woodpecker is computed by applying Equation to the full matrix ***X*** in a single vectorized operation, followed by thresholding and comparison against the ground truth.

4. Position Update: At each iteration, every woodpecker updates its position according to :

$$X_i \leftarrow X_i + \alpha \cdot (X_{best} - X_i) + \varepsilon$$

Where ***Xbest*** is the current best solution, **α** decreases linearly from 1 to 0 across iterations (favoring exploration early and exploitation later), and **ε** is a small random perturbation.

5. Repair and Greedy Selection: After each update, the repair operator restores the normalization and threshold constraints.The updated solution replaces the previous one only if its fitness improves.

6. Termination: Steps 3–5 repeat for a fixed number of iterations.The best solution (***w*, T****) is then reduced, for each attribute, to its single dominant similarity measure, yielding the final matching configuration. The overall workflow is summarized in ***Figure 3.2***.

3.4 Development Environment

For the development of our Record Linkage application, we used the following tools, technologies, and hardware environment:

Hardware Environment

The implementation and experimental evaluation of the proposed framework were carried out on a laptop with the following specifications:

- **Device Model:** Dell Latitude 5410
- **Processor:** Intel® Core™ i5-10210U CPU @ 1.60 GHz (up to 2.11 GHz)
- **Memory (RAM):** 8 GB DDR4 @ 3200 MHz
- **Graphics:** Intel® UHD Graphics
- **Storage:** 238 GB SSD

These hardware specifications provided a suitable environment for data processing, algorithm implementation, and experimental evaluation.

Software Environment

- **Programming Language:** Python (Python 3.14.0). Python was selected because of its simplicity, reliability, and extensive collection of libraries, which facilitate data processing and the implementation of complex algorithms required for Record Linkage.
- **Integrated Development Environment (IDE):** Visual Studio Code (VS Code). VS Code was chosen for its advanced features such as code auto-completion, integrated debugging tools, and extensions that improve development efficiency and productivity.
- **Dataset:** The dataset utilized in this research is **the Restaurant** dataset, which consists of 864 records organized across **8** distinct attributes, with an additional ninth column serving as the ground truth. This dataset contains **112** confirmed identity matches. It is characterized by its heterogeneity, reflecting real-world inconsistencies such as typographical errors and variations in formatting, making it an ideal benchmark to evaluate the robustness of our **ABC** based blocking and **WPO** based matching mechanisms**.**

- **Web Interface:** In addition, we developed a web interface to provide a more user-friendly and attractive environment for interacting with the application. The web interface improves usability by allowing users to easily upload datasets, perform Record Linkage operations, and visualize the results in a clear and organized manner.
- **Python Libraries and Frameworks:** Several Python libraries were used during the development and implementation of the proposed Record Linkage framework:

  - **Streamlit:** Used for developing the web-based user interface.
  - **Pandas:** Used for data manipulation, cleaning, and preprocessing.
  - **NumPy:** Used for numerical computations and array operations.
  - **Plotly:** Used for interactive data visualization and result presentation.
  - **OpenPyXL:** Used for reading and writing Excel files.
  - **SciPy:** Used for scientific computing and optimization-related operations.
  - **Jellyfish:** Used for string similarity measures and phonetic encoding techniques such as Soundex and NYSIIS.
  - **KModes:** Used for clustering categorical data and block optimization.

These libraries significantly facilitated the implementation of the proposed framework and improved both development efficiency and system performance.

3.5 Screenshots

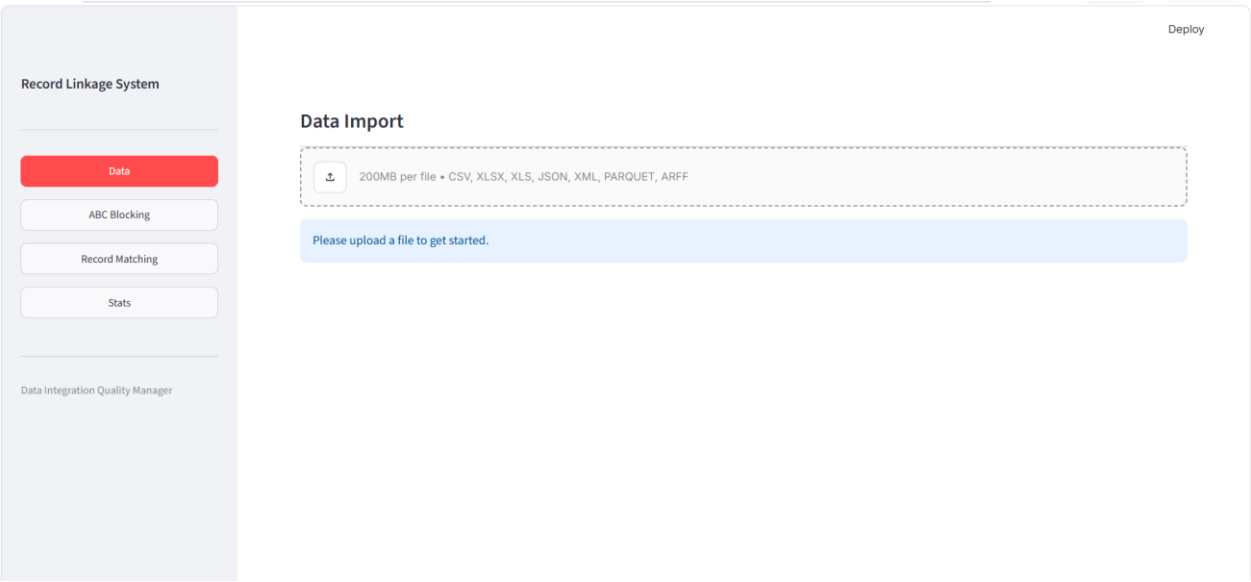

Figure 3.3: Data Import Interface

Description:

This interface represents the initial phase of the system where users can upload heterogeneous datasets. It supports multiple standard data formats (such as CSV, XLSX, and JSON) with a flexible file size capacity. The fixed sidebar establishes the logical workflow sequence of the application.

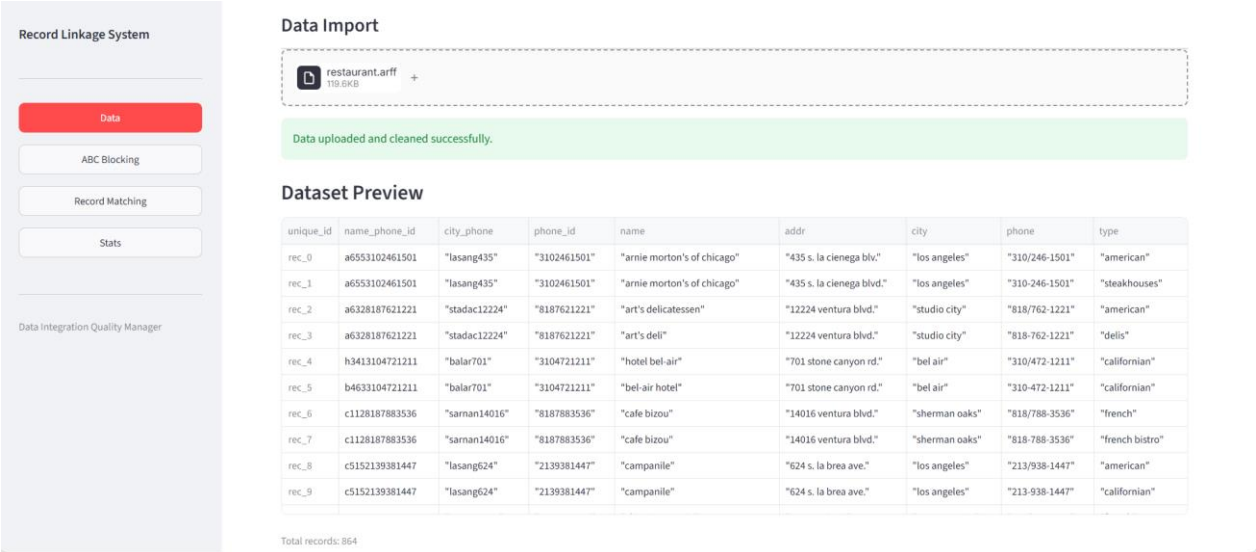

Figure 3.4: Dataset Preview and Verification Interface

Description:

This interface displays the system state after a successful data upload and preliminary cleaning. It provides an interactive preview of the integrated dataset records along with its attributes, and explicitly shows the total workload scale (864 records) to be processed in the subsequent pipeline stages.

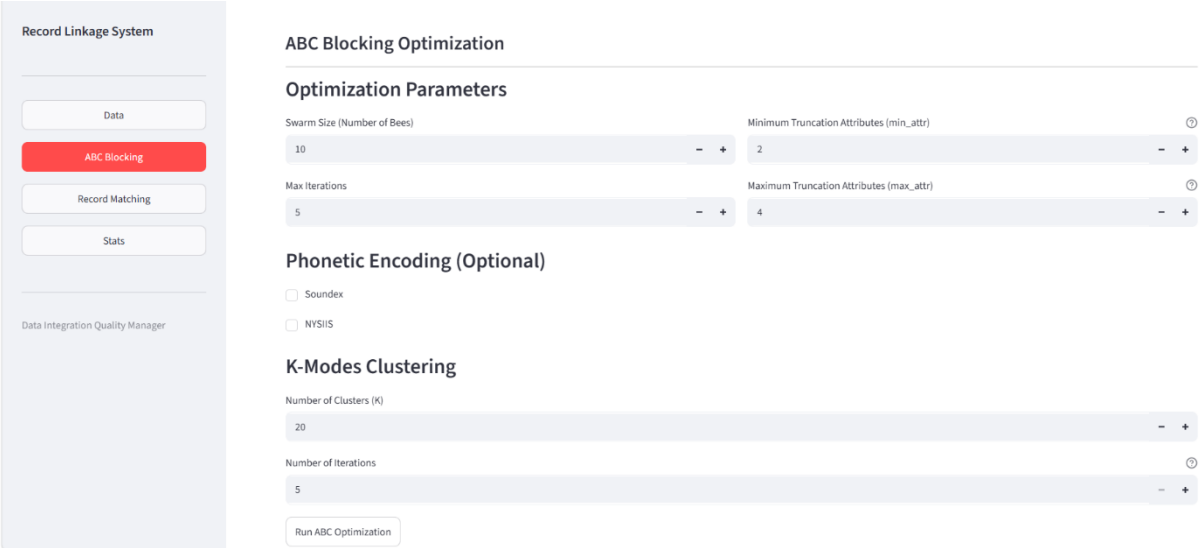

Figure 3.5: ABC Blocking Optimization and Configuration Interface

Description:

This interface allows configuring the hyper-parameters for the implementation pipeline, specifically the Artificial Bee Colony (ABC) optimization parameters (such as Swarm Size and Max Iterations) along with K-Modes Clustering settings. It also includes configuration options for optional Phonetic Encoding algorithms to optimize the generation of the blocking scheme.

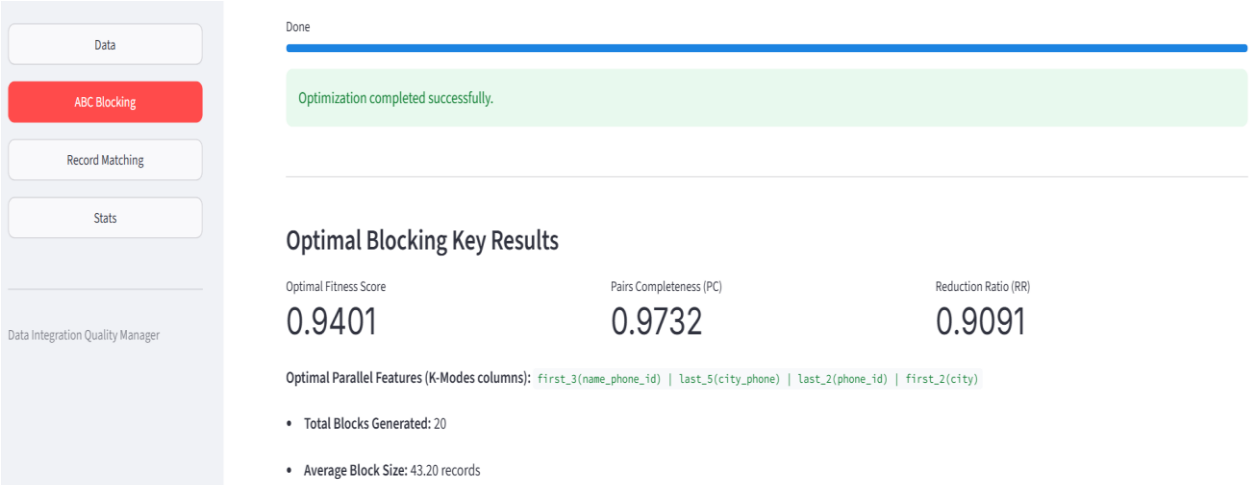

Figure 3.6: Optimal Blocking Results Interface

Description:

This interface presents the execution summary and performance metrics after completing the ABC blocking optimization. It displays key academic evaluation scores, including Pairs Completeness (PC), Reduction Ratio (RR), and the final Optimal Fitness Score, alongside the generated blocking key features and block configuration metrics.

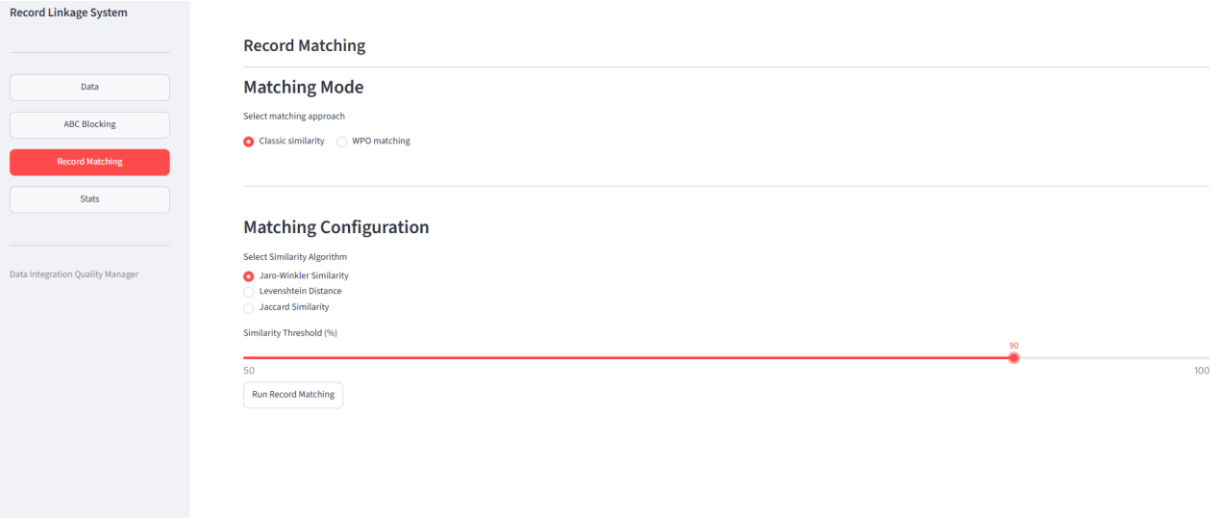

Figure 3.7: Record Matching Configuration Interface

Description:

This interface provides the setup options for the record matching layer. It allows the user to select the preferred execution mode (Classic Similarity vs. WPO Matching) and choose the appropriate core metric (such as Jaro-Winkler, Levenshtein, or Jaccard), while offering an interactive slider to tune the decision threshold before running the matching process.

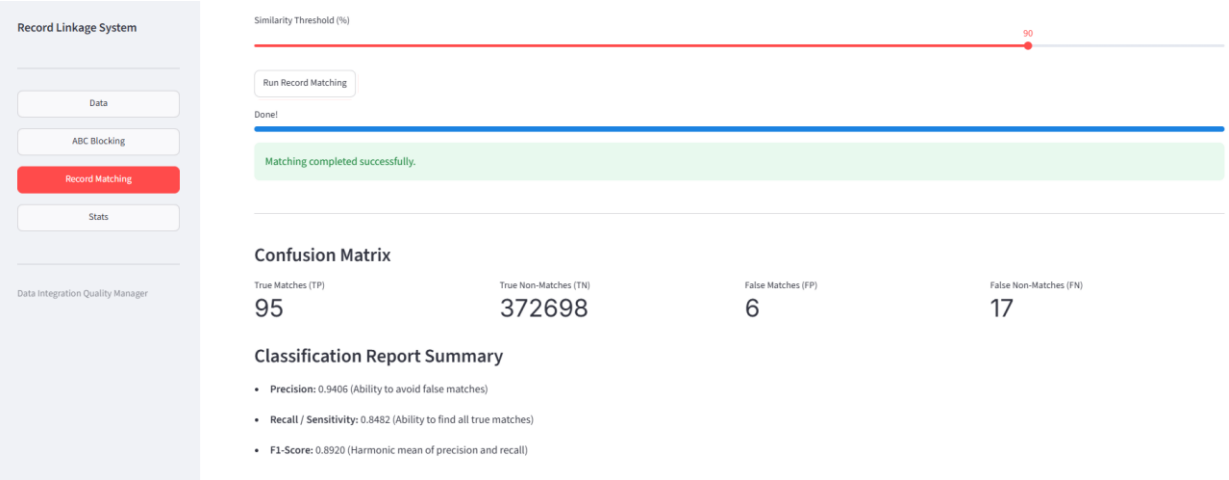

Figure 3.8: Record Matching Execution and Evaluation Interface

Description:

This interface displays the immediate evaluation results following the execution of the record matching process. It presents the complete Confusion Matrix parameters (TP, TN, FP, FN) along with a structured classification report summary containing standard evaluation metrics (Precision, Recall, and F1-Score) to assess the classification accuracy.

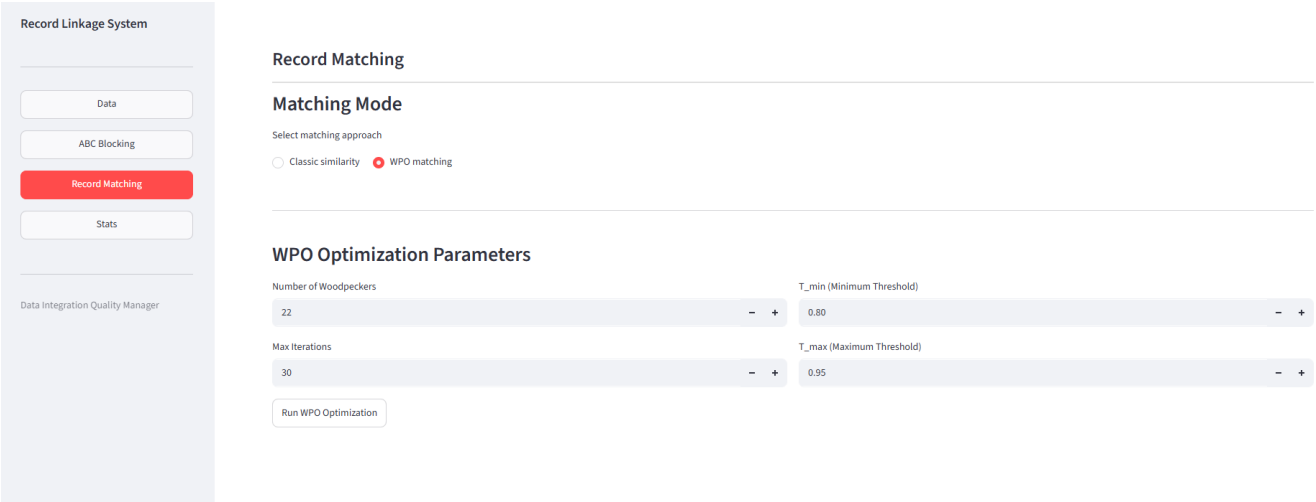

Figure 3.9: WPO Metaheuristic Configuration Interface

Description:

This interface comes into view when selecting the "WPO Matching" mode, allowing users to configure the optimization hyper-parameters for the Woodpecker Optimization Algorithm. It provides configuration counters for the population size (Number of Woodpeckers), maximum iterations, and boundary search thresholds ***Tmin***, ***Tmax*** before running the weight optimization process.

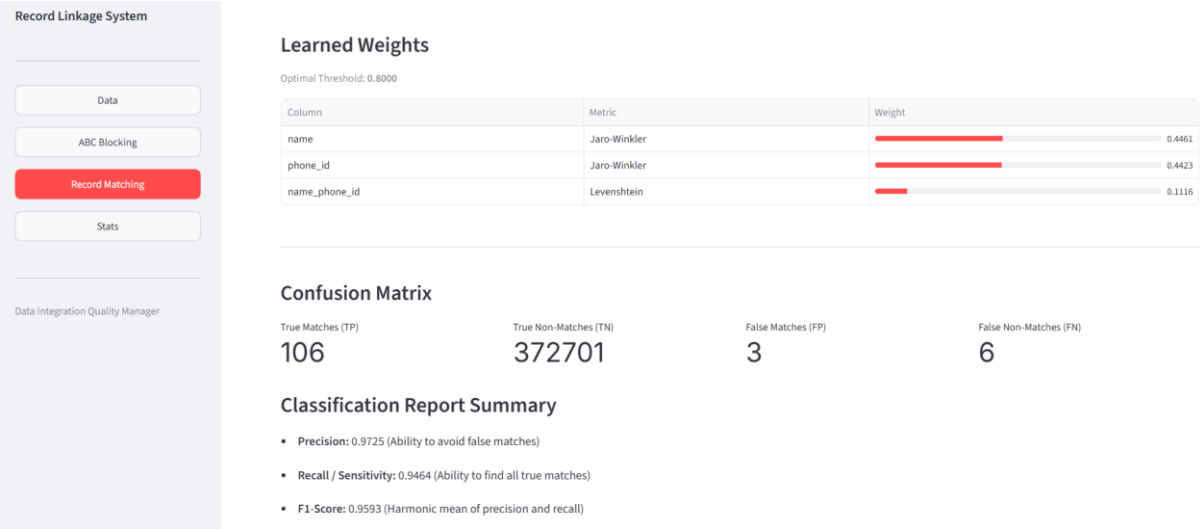

Figure 3.10: Optimized Weight Learning and Evaluation Interface

Description:

This interface displays the final outputs of the optimized WPO matching mode. It features the "Learned Weights" table, where each attribute is selectively mapped to its single best-performing similarity metric along with its normalized weight. It also provides the enhanced Confusion Matrix and a high-accuracy classification report summary reflecting the updated pipeline state.

Figure 3.11: System Analytics Interface and Performance Graphical Representation

Description:

This dashboard interface provides a comprehensive statistical analysis of the system's final matching performance. It consolidates the global Confusion Matrix and classification values alongside an interactive bar chart that visually compares key performance indicators (Precision, Recall, and F1-Score). It also includes an integrated definitions section for metrics interpretation.

3.6 Evaluation and Experimental Results:

3.6.1 *Evaluation*

In the Record Linkage community, four main parameters are used to measure the performance of an RL process.

1- Precision: This metric is used to measure the precision of comparisons.

$$Precision = \frac{Tp}{Tp + Fp}$$

2- Recall: This metric is used to measure the ratio of correctly predicted links from all true matches.

$$Recall = \frac{Tp}{Tp + Fn}$$

3- F-Score: The F-Score is used to measure the harmonic mean between the two previous parameters.

$$F\text{-}Score = \frac{2 \times Precision \times Recall}{Precision + Recall}$$

Where:

- **Tp (True Positives):** Pairs that appear in the same cluster in both the ground truth and the prediction. Known as true matches.
- **Tn (True Negatives):** Pairs that appear in different clusters in both the ground truth and the prediction. Known as true non-matches.
- **Fp (False Positives):** Pairs that appear in the same cluster in the prediction but in different clusters in the ground truth. Known as false matches.

Fn (False Negatives): Pairs that appear in the same cluster in the ground truth but in different clusters in the prediction. Known as false non-matches or missed matches.

3.6.2 Experimental Results

3.6.2.1 Results: This section presents a comparative performance analysis of the proposed Woodpecker Optimization Algorithm (WPO) matching strategy against standard individual similarity metrics. To ensure a fair and rigorous benchmark, each method was evaluated at its respective empirically determined optimal threshold (***T***).

Metric	Jaro-Winkler (T=92%)	Levenshtien(T=93%)	Jaccard(T=60%)	WPO (T=80%)
TP	**104**	**102**	**93**	**106**
FP	**3**	**2**	**3**	**3**
FN	**8**	**10**	**19**	**6**
TN	**372,701**	**372,702**	**372,701**	**372,701**
Precision	**0.9720**	**0.9808**	**0.9688**	**0.9725**
Recall	**0.9286**	**0.9107**	**0.8304**	**0.9464**
F&-Score	**0.9498**	**0.9444**	**0.8942**	**0.9593**

Table 3.1: Comparison of Matching Strategies at Optimal Thresholds

3.6.2.2 Discussion and Analysis:

The results in **Table 3.1** demonstrate the superiority of the proposed WPO framework over individual similarity metrics. While the Levenshtein distance achieves slightly higher precision (**0.9808**), it suffers from a degraded recall. In contrast, WPO successfully minimizes missed matches (recording the lowest **FN = 6**) and unlocks the highest recall (**0.9464**). Ultimately, the WPO approach provides the best classification balance, outperforming all baseline methods with a peak F1-Score of **0.9593** at the dynamically discovered optimal threshold of **T = 0.80**.

3.7 Conclusion

In conclusion, this chapter successfully demonstrated the design and implementation of the developed Record Linkage System. By transitioning to a web-based Python application, the system delivers an intuitive pipeline that integrates dataset ingestion, evolutionary ABC blocking optimization, and advanced WPO feature-weight matching. The experimental results validate that combining these metaheuristic algorithms drastically reduces the computational search space while achieving high accuracy, as confirmed by the optimal precision, recall, and F1-score metrics.

General Conclusion

In today's data-driven world, the continuous growth of information generated from multiple sources has made data quality a major challenge for organizations and researchers. Among the various techniques used to improve data quality, Record Linkage plays a crucial role in identifying duplicate records and integrating information referring to the same real-world entity. However, the Record Linkage process becomes computationally expensive when applied to large datasets due to the large number of potential record comparisons. Consequently, improving the efficiency of both the blocking and matching phases is essential for achieving accurate and scalable Record Linkage systems. In this dissertation, we proposed a Record Linkage framework based on metaheuristic optimization techniques. During the blocking phase, the Artificial Bee Colony (ABC) algorithm was employed to automatically select the most relevant blocking keys. The selected blocking keys were then used to generate candidate blocks while maximizing Pair Completeness and reducing unnecessary comparisons. To address the imbalance observed in the generated blocks, K-Modes clustering was integrated into the framework. This additional step helped create more balanced blocks, reduce excessively large blocks, and improve the overall distribution of records before the matching phase. Furthermore, the matching phase was optimized using the Woodpecker Optimization Algorithm (WPO). The algorithm was applied to automatically determine the optimal similarity weights and matching threshold, allowing a better balance between Precision, Recall, and F1-Score. As a result, the proposed approach improved duplicate detection accuracy while reducing the need for manual parameter tuning. The obtained results demonstrate that the combination of ABC, K-Modes, and WPO contributes significantly to improving the efficiency, scalability, and effectiveness of the Record Linkage process. The proposed framework successfully reduces the search space, minimizes unnecessary comparisons, and enhances matching quality. As future work, the proposed approach can be evaluated on larger and more diverse datasets. Additional similarity measures and alternative metaheuristic optimization algorithms may also be investigated to further improve matching performance and scalability.

General Conclusion

In conclusion, this work demonstrates that combining intelligent optimization techniques with Record Linkage provides an effective solution for improving data quality and duplicate detection in large-scale data integration environments.

Bibliography

[1]Assia Soukane. Génération automatique des requêtes de médiation dans un environnement hétérogène. PhD thesis, Versailles-St Quentin en Yvelines, 2005.

[2] Isabel F Cruz and Huiyong Xiao. The role of ontologies in data integration. Engineering intelligent systems for electrical engineering and communications, 13(4) :245, 2005.

[3] Dung Xuan Nguyen. Intégration de bases de données hétérogènes par articulation à priori d'ontologies : application aux catalogues de composants industriels. PhD thesis, Université de Poitiers, 2006.

[4] Gema Bello-Orgaz, Jason J Jung, and David Camacho. Social big data: Recent achievements and new challenges. Information Fusion, 28:45–59, 2016.

[5] Arvind Sathi. Big data analytics. Mc Press, 2012.

[6] Louardi Bradji and Mahmoud Boufaida. Adaptation des techniques de l'extraction des connaissances à partir des données (ecd) pour prendre en charge la qualité des données. 2017.

[7] Abdelkrim OUAHAB. Qualité de données pour l'intégration de données. PhD thesis, Université de Sidi Bel Abbès-Djillali Liabes, 2019

[8]Franck Rgnier-Pcastaing, Michel Gabassi,
Jacques Finet,Enjeux et mthodes dela gestion
des donnes,livre,(2008).

[9] JEMM research, DES DONNES QUALIT

:Exploitez lecapital de votre organisa-tion

,livre blanc,(janvier 2008).

[10] Laure Berti-Équille. La qualité des données comme condition à la qualité des connaissances: un état de l'art. Revue des Nouvelles Technologies de l'Information, 2004.

[11] Laure Berti-quille, Qualit des donnes,Matre de Confrences

[12] Maurizio Lenzerini. Data integration : A theoretical perspective. pages 233–246, 2002.

[13] Mohand-Said Hacid and Chantal Reynaud. L'intégration de sources de données. Revue

Information-Interaction-Intelligence, 3 :4, 2004.

[14] Synology Inc. (2022). *Livre blanc sur la déduplication des données : Basé sur Active Backup for Business 2.3.0* [White Paper]. Synology. https://www.synology.com

[15] Laure Berti-Equille. Qualité des données.

Techniques de l'ingénieur.Informatique,

2006.

[16] https://recordlinkage.readthedocs.io/en/latest/about.html

[17] Abdelkrim OUAHAB. Qualité de données pour l'intégration de données. PhD thesis,

Université de Sidi Bel Abbès-Djillali Liabes, 2019.

[18]Peter Christen. A survey of indexing techniques for scalable record linkage and deduplication. IEEE transactions on knowledge and data engineering, 24(9):1537–1555,

2011

[19] David E Clark. Practical introduction to record linkage for injury research. Injury

Prevention, 10(3):186–191, 2004.

[20] Peter Christen. A survey of indexing techniques for scalable

record linkage and deduplication. IEEE transactions on

knowledge and data engineering, 24(9) :,1537–1555,, b.

Peter Christen. Towards parameter-free blocking for scalable record linkage

[21] Nascimento D.C, Pires C.E.S., et Mestre D.G. Exploiter la cooccurrence de bloc pour contrôler la taille des blocs pour la résolution d'entité. Knowledge and Information Systems, 62(1), 359- 400, 62(1) : 359–400.

[22] Chandrashekar G et Sahin F. An introduction to variable and feature selection, guyon, isabelle and elisseeff, andré,. Journal of machine learning research, 3(1) :16–28. numéro : mars, pages : 1157–1182, année : 2003. Informatique et génie électrique.

[23] TN Gadd. Phonix : The algorithm. Program, 24(4) :363–366,. Köpcke H, Thor A, et Rahm E. Evaluation of entity resolution approaches on real-world match problems. Proceedings of the VLDB Endowment, 3(1-2) :484–493.

[23] Köpcke H, Thor A, et Rahm E. Learningbased approaches for matching web data entities. IEEE Internet Computing, 14(4) :23–31, b. Mauricio A Hernàndez et Salvatore J Stolfo. The merge/purge problem for large databases. Dans ACM Sigmod Record, volume 24, page 127–138. ACM.

[24] TN Gadd. Phonix : The algorithm. Program, 24(4) :363– 366,. Köpcke H, Thor A, et Rahm E. Evaluation of entity resolution approaches on real-world

[25] David Holmes et M.Catherine McCabe. Improving precision and recall forsoundex retrieval. Dans Information Technology : Coding and

Computing, 2002.Proceedings. International Conference on, page 22–
26. IEEE.

[26] Brezočnik, L., Fister, I., & Podgorelec, V. (2018). Swarm intelligence algorithms for feature selection: A review. Applied Sciences, 8(9), 1521. https://doi.org/10.3390/app8091521

[27] Karaboğa, D. (2005). An idea based on honey bee swarm for numerical optimization (Technical Report TR06). Erciyes University, Engineering Faculty, Computer Engineering Department.

[28] Karaboğa, D., & Akay, B. (2009). A comparative study of Artificial Bee Colony algorithm. Applied Mathematics and Computation, 214(1), 108–132. https://doi.org/10.1016/j.amc.2009.03.090

[29] Karaboğa, D., & Basturk, B. (2007). A powerful and efficient algorithm for numerical function optimization: Artificial Bee Colony (ABC) algorithm. Journal of Global Optimization, 39(3), 459–471. https://doi.org/10.1007/s10898-007-9149-x

[30] Nguyen, B. H., Xue, B., & Zhang, M. (2020). A survey on swarm intelligence approaches to feature selection in data mining. Swarm and Evolutionary Computation, 54, 100663. https://doi.org/10.1016/j.swevo.2020.100663

[31] Pipino, L. L., Lee, Y. W., & Wang, R. Y. (2002). Data quality assessment. Communications of the ACM, 45(4), 211–218. https://doi.org/10.1145/505248.506010

[32] Tereshko, V., & Loengarov, A. (2005). Collective decision-making in honey-bee foraging dynamics. Computing and Information Systems, 9(3), 1–7. PartagerContenupdf.

[33] Huang, Z. (1998). Extensions to the k-means algorithm for clustering large data sets with categorical values. Data Mining and Knowledge Discovery, 2(3), 283–304. https://doi.org/10.1023/A:1009769707641

[34] Huang, Z. (1997). A Fast Clustering Algorithm to Cluster Very Large Categorical Data Sets in Data Mining. Proceedings of the SIGMOD Workshop on Research Issues on Data Mining and Knowledge Discovery.

[35] Mirjalili, S., Mirjalili, S. M., & Lewis, A. (2014). Grey Wolf Optimizer. Advances in Engineering Software, 69, 46–61. https://doi.org/10.1016/j.advengsoft.2013.12.007

[36] Karaboga, D. (2005). An idea based on honey bee swarm for numerical optimization. Technical Report-TR06, Erciyes University.

[37] Short, L. L. (1982). Woodpeckers of the World. Delaware Museum of Natural History.