Web Of Data: How to publish Linked Data

Pablo Mollá Chárlez

Contents

1 Introduction

A **knowledge graph (KG)** is a structured representation of information, where **entities** (such as people, places, and objects) are **linked by relationship**s, providing a **semantic framework** to capture complex knowledge. By using RDF (Resource Description Framework) and linked data principles, knowledge graphs enable seamless data integration across diverse sources, allowing entities to be interconnected in a meaningful way. They **support query capabilities through SPARQL**, making it easy to retrieve information based on relationships and attributes. The **ontology hierarchy** organizes entities into classes and subclasses, like dbo:Museum and dbo:PopulatedPlace, while ontology axioms and rules, defined in OWL (Web Ontology Language), establish constraints and equivalencies.

Many companies have adopted these technologies for enhanced data management and insights, as shown by the broad adoption of knowledge graphs across industries. Organizations such as **Google**, **Facebook**, **Amazon**, and **Microsoft** leverage knowledge graphs and linked data to power their search, recommendation, and decision-making systems. These tools are also widely used by major research and public institutions, including **PubMed**, the **Library of Congress**, and the **European Commission**, which benefit from improved data accessibility, interoperability, and enriched semantic relationships among data points.

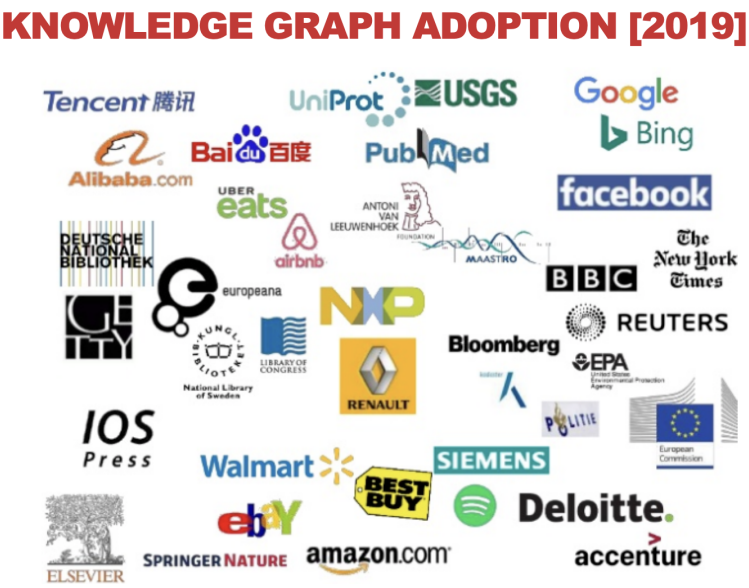

Figure 1: Knowledge Graph Adoption

2 5 Star-Schema of Linked Data

The **5-Star Linked Data Schema** is a framework introduced by **Tim Berners-Lee** to guide the publishing of data on the Web in a way that maximizes its accessibility, interoperability, and reusability. Each "star" represents a level of maturity in how the data is published, with higher stars indicating more robust and interconnected data suitable for building comprehensive knowledge graphs. A detailed explanation of each level is described as follows:

1. **1 Star: Make Your Data Available on the Web** ☆

   At the most basic level, data should be accessible on the Web under an open license. This means ensuring that anyone can find and access the data, regardless of the format it is in.

   Example: Publishing a PDF report containing statistical data about city demographics on your website with a clear open license.

2. **2 Stars: Structure Your Data** ☆☆

   Enhance the data's usability by providing it in a structured format (formats like CSV, JSON, XML, or Excel instead of unstructured formats like images or PDFs). Structured data (into tables, records, and fields to facilitate querying and analysis) allows machines to parse and interpret the data more easily.

   Example: Providing the same city demographics data in a CSV file where each row represents a different city and each column represents a specific demographic attribute (e.g., population, median income).

3. **3 Stars: Use Non-Proprietary Formats** ☆☆☆

   Ensure the data is available in open, non-proprietary formats (prefer formats like CSV, JSON, XML, or RDF over proprietary formats like Excel (.xlsx)) to promote long-term accessibility and avoid vendor lock-in.

   Example: Choosing to publish demographic data in CSV format instead of an Excel spreadsheet to ensure that users can access the data without needing specific software.

4. **4 Stars: Use URIs to Identify Entities** ☆☆☆☆

   Implement Uniform Resource Identifiers (URIs) to uniquely identify entities within the data. This enables distinct entities to be referenced unambiguously on the Web (allowing interlinked data).

   Example: Assigning a URI like **http://example.org/cities/London** to uniquely identify the city of London, enabling others to reference and connect data about London from various sources.

5. **5 Stars: Link Your Data to Other Data** ☆☆☆☆☆

   Enhance the data's context and value by linking it to other relevant datasets on the Web by using Linked Data Principles. This creates a web of interconnected data, forming a foundation for comprehensive knowledge graphs.

   Example: Linking your city demographics data to other datasets such as geographic information systems (GIS) data, economic indicators, or cultural resources by referencing their URIs. For instance, linking **http://example.org/cities/London** to **http://dbpedia.org/resource/London** from DBpedia.

Putting it all together, we build a Knowledge Graph with the following properties: **Availability**, **Structure**, **Openness**, **Identification**, **Linking**.

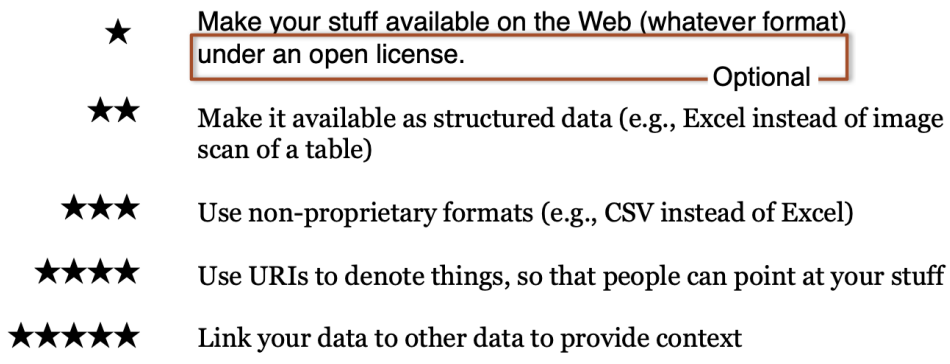

Figure 2: 5 Star-Schema of Linked Data

2.1 Benefits of Linked Data

1. **Enhanced Discoverability**: Structured and linked data is easier to find and use.
2. **Improved Interoperability**: Non-proprietary formats and URIs facilitate seamless data integration.

3. **Greater Reusability**: Open licenses and interconnected data allow for broader applications and innovations.

4. **Rich Knowledge Graphs**: Linking data creates a web of information that can support complex queries and insights.

3 The need of Knowledge: Ontologies

While data represents raw facts and figures, knowledge involves the interpretation, relationships, and context that give data meaning. To transition from mere data to actionable knowledge, especially in complex domains like knowledge graphs, we need a structured framework to define and manage this knowledge systematically. This is where ontologies come into play.

3.1 Understanding Ontologies

Thomas R. Gruber (1993) defined an ontology as:

> "An ontology is an explicit, formal specification of a shared conceptualization."

Let's unpack this definition to outline the main components of an ontology:

1. **Conceptualization**: An abstract model that outlines the key concepts and relationships within a domain. For example, in the art domain, concepts might include yellowArtistsyellow, yellowArtworksyellow, yellowMuseumsyellow, etc.

2. **Specification**: A detailed description tailored to a specific domain. It defines the exact meaning and constraints of each concept and relationship.

3. **Explicitness**: All terms and their meanings are clearly defined. There's no ambiguity; every expression's semantics are transparent.

4. **Formality**: The ontology is created using formal languages (like OWL) that machines can interpret. Enables automated reasoning and validation.

5. **Shared**: Represents a consensus view, ensuring that different stakeholders have a unified understanding. Facilitates interoperability and data integration across different systems.

3.2 OWL Ontology and Reasoning

OWL (Web Ontology Language) is a powerful language used to create and manage ontologies. It supports complex relationships and reasoning, enabling machines to infer new knowledge from existing data. It contains Class Hierarchies, Axioms, Instances (individual definitions) and Property definitions. The benefits of the inference of data or reasoning are:

- **Consistency Checking**: Ensures that the data adheres to the defined ontology rules.
- **Knowledge Expansion**: Automatically infers new facts without manual input.
- **Enhanced Querying**: Allows more sophisticated queries based on inferred knowledge.

3.3 Linked Data Ontologies

Linked Data Ontologies are essential frameworks that provide common vocabularies enabling search engines and machines to understand and process data embedded within web pages. By incorporating micro-data directly into HTML, these ontologies facilitate a web environment that is navigable and interpretable by both humans and machines alike. The key goals of Linked Data Ontologies are:

- **Dual Accessibility**: Create aweb that serves the needs of both human users and automated systems, enhancing **usability** and **functionality**.
- **Standardized Metadata**: Empower webmasters to embed metadata using established web standards and structured HTML, ensuring **consistency** and **interoperability**.
- **Semantic Understanding**: Enable access to the underlying meaning of web content, allowing for more intelligent data processing and retrieval.
- **Data Interconnectivity**: Establish meaningful relationships between disparate data sources, promoting seamless exploration and discovery across different datasets.

4 Ontology Alignment: Part I

Ontology Alignment is a critical process in the realm of semantic web and knowledge graphs, aimed at achieving interoperability between different ontological frameworks. Essentially, ontology alignment involves matching and merging concepts, relationships, and structures from two distinct ontologies to establish a coherent and unified understanding across diverse datasets.

4.1 Core Components of Ontology Alignment

Performing ontology alignment necessitates the following concepts to be done successfully:

1. **Source Ontologies (θ_1 and θ_2)**: These are the two separate ontological frameworks that need to be aligned. Each ontology defines its own set of concepts, relationships, and rules within a specific domain.
2. **Alignment Process**: The primary goal is to create an alignment (A') that maps corresponding elements between θ_1 and θ_2. This involves identifying equivalent or related concepts and relationships to ensure seamless data integration and interoperability.
3. **Input Alignment (A) and External Resources**: Optionally, the alignment process can utilize an existing input alignment (A), which serves as a preliminary mapping between the ontologies.
4. **Alignment Relations (θ)**: The resulting alignment (A') consists of pairs of elements from θ_1 and θ_2 connected by specific alignment relations. These relations define the nature of the correspondence and can include: **owl:equivalentClass** (indicates that two classes from different ontologies are equivalent), **rdfs:subClassOf** (denotes a hierarchical relationship where one class is a subclass of another), **owl:disjointWith** (specifies that two classes are mutually exclusive), **closeTo** (represents a near-equivalence or partial match between concepts), etc.

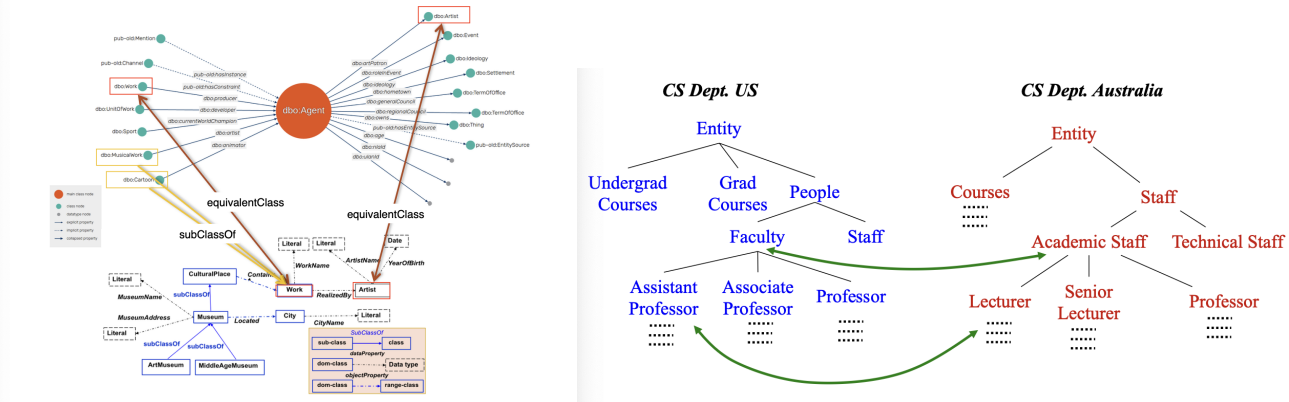

Figure 3: Museum Ontology & Computer Science Department

4.2 Problem of Heterogeneity

Heterogeneity is a fundamental challenge in ontology alignment, arising from the differences between various models that represent the same or overlapping domains. These differences manifest in several key areas:

1. **Coverage**: Coverage refers to the extent to which two ontologies describe the same domain but share a more or less important set of concepts and properties.
2. **Granularity**: Granularity deals with the level of detail each ontology provides within the same domain, where a given ontology can range from highly detailed, catering to experts, and another to more generalized versions suitable for broader audiences.
3. **Different Points of View** Ontologies may represent the same domain from distinct perspectives or frameworks, leading to varied interpretations and relationships between concepts.

4.3 Similarity Measures: Ontologies

After understanding the conceptual heterogeneity challenges in ontology alignment—such as differences in coverage, granularity, and points of view—we turn to similarity measures as essential tools to bridge these gaps. Similarity measures quantify the likeness between elements (e.g., classes, properties) of different ontologies, facilitating the alignment process by identifying potential matches despite underlying differences. We will consider 3 types of possible similarity measures, each type offers distinct advantages for handling various aspects of conceptual heterogeneity. including: **Token Based**, **Edit Based** and **Hybrids**.

4.3.1 Token Based

Token-based measures evaluate similarity based on the presence and frequency of tokens (words or terms) within the elements being compared. They are particularly useful for handling cases where different ontologies use varying terminology to describe similar concepts.

4.3.1.1 Jaccard Similarity

Measures the overlap between two sets of tokens relative to their union and is defined as:

$$\text{Jaccard}(S,T) = \frac{|S \cap T|}{|S \cup T|}$$

4.3.1.2 Cosine Similarity (Based on TF-IDF)

Calculates the cosine of the angle between two term frequency-inverse document frequency (TF-IDF) vectors, emphasizing rare but significant terms.

$$\text{Cosine}(S,T) = \frac{\sum_{i=1}^{n} \text{TF-IDF}(s_i) \times \text{TF-IDF}(t_i)}{\sqrt{\sum_{i=1}^{n} \text{TF-IDF}(s_i)^2} \times \sqrt{\sum_{i=1}^{n} \text{TF-IDF}(t_i)^2}}$$

Where $\text{TF}(t,S)$ = Term Frequency of term t in string S and $\text{IDF}(t) = \log\left(\frac{\text{Total number of documents}}{\text{Number of documents containing } t}\right)$.

The advantages include that highlights important terms and is efficient for large datasets, however it's sensitive to spelling errors and abbreviations, and ignores the order of words.

4.3.2 Edit Based

Edit-based measures assess similarity based on the minimum number of edit operations required to transform one string into another. They are effective for handling typographical errors and slight variations in terminology.

4.3.2.1 Levenshtein Distance

The Levenshtein distance computes the minimum number of single-character edits (insertions, deletions, substitutions) needed to change one string into another. Let $\text{lev}(s,t)$ represent the Levenshtein distance between strings s and t. The distance is calculated using the following recurrence relation:

$$\text{lev}(s,t) = \begin{cases} |t| & \text{if } |s| = 0 \\ |s| & \text{if } |t| = 0 \\ \text{lev}(s_{1..m-1}, t_{1..n}) + 1 & \text{if } s[m] \neq t[n] \\ \min \begin{cases} \text{lev}(s_{1..m-1}, t) + 1 \\ \text{lev}(s, t_{1..n-1}) + 1 \\ \text{lev}(s_{1..m-1}, t_{1..n-1}) + 1 \end{cases} & \text{if } s[m] \neq t[n] \end{cases}$$

Where $s[m]$ and $t[n]$ are the last characters of strings s and t, respectively.

For instance, s = **"Error"** $\rightarrow$ t = **"Eror"**, then $lev(s,t) = 1$ (corresponding to the substitution of **'r'** with **'o'**). The advantages include the handling of minor variations and typos, although it becomes computationally intensive for long strings and does not capture semantic similarity beyond character edits.

4.4 Hybrid Similarity Measures

Hybrid measures combine elements of both token-based and edit-based approaches to leverage their respective strengths, providing a more comprehensive similarity assessment.

4.4.0.1 Jaro Similarity

Focuses on the number and order of matching characters between two strings s_1 and s_2, giving higher scores to strings with more matching characters in similar order.

$$\text{Jaro}(s,t) = \frac{1}{3}\left(\frac{m}{|s_1|} + \frac{m}{|s_2|} + \frac{m-t}{m}\right)$$

Where the parameters m is the number of matching characters and t the number of transpositions.

4.4.0.2 Jaro-Winkler Similarity

This similarity measure enhances **Jaro Similarity** by giving additional weight to matching characters that appear at the beginning of the strings, improving performance on common typographical errors and prefixes.

$$\text{Jaro-Winkler}(s,t) = \text{Jaro}(s,t) + (l \cdot p \cdot (1 - \text{Jaro}(s,t)))$$

Where l is the length of the common prefix at the start of the string (up to a maximum of 4 characters) and p the scaling factor (typically 0.1). The main drawback is that is more complex to implement than purely token-based or edit-based measures, and may struggle with significant semantic differences, however, it balances character matching with positional information and becomes more resilient to transpositions and common spelling mistakes.

5 Ontology Alignment: Part II

Building upon the understanding of conceptual heterogeneity and the role of similarity measures in ontology alignment, we now delve into how these measures can be specifically applied to the internal and external structures of ontologies. This distinction allows for **a more nuanced and effective alignment process** by **addressing both the inherent properties of ontology** elements and their **interrelationships**.

5.1 Similarity of Internal Structures

Internal structure pertains to the specific attributes and constraints that define ontology elements, such as datatype properties, ranges, and cardinalitie.

- **Datatype Compatibility**: Assess how compatible different datatypes are between ontologies.
- **Cardinality Comparison**: Evaluate the similarity of cardinality constraints using the following OWL-based formula:

$$\text{Sim}([b1, e1], [b2, e2]) = \begin{cases} 0 & \text{if } b2 > e1 \text{ or } b1 > e2 \\ \frac{\min(e1,e2) - \max(b1,b2)}{\max(e1,e2) - \min(b1,b2)} & \text{otherwise} \end{cases}$$

5.2 External Structure Similarity

External structure involves the relationships and linked concepts within the ontology, focusing on how concepts are interconnected.

There is a **main hypothesis** used for studying the similarity of external structures within an ontology, which is the more similar two concepts are, the more similar their linked concepts will be. For instance, the similarity Relation-Based Measures can be studied as follows:

1. **Direct Links** (r): Immediate relationships, such as 'subclassOf'.
2. **Transitive Closure** (r^+): All concepts connected through a transitive property.
3. **Inverse Relations** (r^{-1}): More general or broader concepts linked inversely.
4. **Leaves** ($r!$): Terminal concepts in the hierarchy.

5.3 External Resource

5.3.1 Wu and Palmer Similarity

The **Wu and Palmer (WUP) Similarity Measure**, introduced by Wu and Palmer in 1994, is a semantic similarity metric used to quantify how similar two concepts (C_1 and C_2) are within a hierarchical ontology, such as WordNet. This measure is particularly useful in **ontology alignment** and **knowledge graph** applications to assess the relatedness of different entities.

$$\text{Sim}(C_1, C_2) = \frac{2 \cdot \text{level}(C)}{\text{level}(C_1) + \text{level}(C_2)}$$

Where the components are:

- **Level(C)**: Represents the depth of the Least Common Subsumer (LCS) in the ontology's hierarchy, an is defined as the most specific concept that is an ancestor of both C_1 and C_2.
- **Level(C_1)** and **Level(C_2)**: Indicate the depth of each individual concept (C_1 and C_2) from the root of the ontology.

For instance, let's consider the following concept θ with its sub-concepts C_1 and C_2 within the ontology θ with 5 levels, starting from **"Computer Science"** (level 1) and finishing in **"Supervised Learning"** and **"Unsupervised Learning"** (both at level 4):

- For $C =$ **"Machine Learning"**, $C_1 =$ **"Supervised Learning"** and $C_2 =$ **"Unsupervised Learning"**, we would have:

$$\text{Sim}(C_1, C_2) = \frac{2 \cdot \text{level}(C)}{\text{level}(C_1) + \text{level}(C_2)} = \frac{2 \cdot 3}{4 + 4} = \frac{6}{8} = \frac{3}{4} = 0.75$$

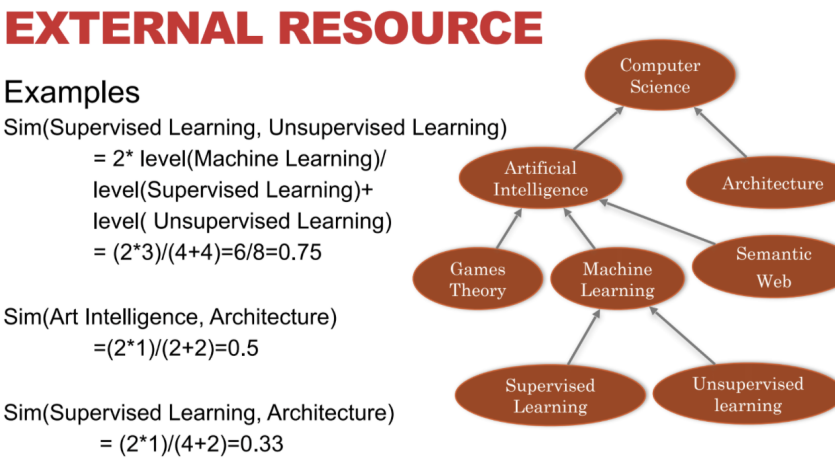

Figure 4: Wu & Palmer Similarity

After computing the **Wu and Palmer Similarity** for each pair of concepts, we obtain the following semantic closeness table:

	CS	AI	Arch	GT	ML	SW	SupL	UnsupL
CS	1.000000	0.666667	0.666667	0.500000	0.500000	0.500000	0.400000	0.400000
AI	0.666667	1.000000	1.000000	0.800000	0.800000	0.800000	0.666667	0.666667
Arch	0.666667	1.000000	1.000000	0.800000	0.800000	0.800000	0.666667	0.666667
GT	0.500000	0.800000	0.800000	1.000000	1.000000	1.000000	0.857143	0.857143
ML	0.500000	0.800000	0.800000	1.000000	1.000000	1.000000	0.857143	0.857143
SW	0.500000	0.800000	0.800000	1.000000	1.000000	1.000000	0.857143	0.857143
SupL	0.400000	0.666667	0.666667	0.857143	0.857143	0.857143	1.000000	1.000000
UnsupL	0.400000	0.666667	0.666667	0.857143	0.857143	0.857143	1.000000	1.000000

5.4 Concept Extensions

It's also possible to use concept instances with extensional information in an ontology, to determine how similar 2 concepts are between each other. We examine the overlap of instances (or examples) belonging to each concept. This overlap can help define relationships such as equivalence, subsumption, and disjunction.

5.4.0.1 Naive Approach

We can use the following relationships to evaluate the similarity:

- **Equivalence**: Two concepts are equivalent if all instances in one concept are also in the other and vice versa. For instances, if all instances in **"Machine Learning"** are the same as those in **"Artificial Intelligence"** (unlikely in practice), we would say they are equivalent.
- **Subsumption**: One concept subsumes another if all instances in the second concept are also in the first. For instances, if all instances in **"Supervised Learning"** are also in **"Machine Learning"** then "Machine Learning" subsumes **"Supervised Learning"**.
- **Disjunction**: Two concepts are disjoint if they have no instances in common. For instances, if **"Games Theory"** and **"Semantic Web"** have no overlapping instances, they are considered disjoint.

Suppose we have the following instance sets for two concepts:

- **Machine Learning (C1)**: $\{Instance1, Instance2, Instance3, Instance4\}$

- **Artificial Intelligence (C2)**: $\{Instance3, Instance4, Instance5, Instance6\}$. Then, we would have:
  - **Intersection** $(C1 \sqcap C2)$: $\{Instance3, Instance4\}$
  - **Union** $(C1 \sqcup C2)$: $\{Instance1, Instance2, Instance3, Instance4, Instance5, Instance6\}$

Using the Jaccard formula:

$$\text{Jaccard}(C1, C2) = \frac{|C1 \cap C2|}{|C1 \cup C2|} = \frac{2}{6} = 0.33$$

This Jaccard similarity score indicates a moderate overlap between **"Machine Learning"** and **"Artificial Intelligence"**, which suggests they share some common instances but also have distinct ones.

However, how can we know whether two instances are the same? The problem of identifying whether two instances are the same can be resolved if they share the same URI or, if they have different URIs, by confirming that they share the same document URL (as in DBpedia/Yago); otherwise, a linking method is needed to detect identity links between them.

5.5 Scalability Strategies

Scalability strategies include computing similarity scores before selecting proposed mappings, combining different strategies in parallel, applying strategies sequentially, and partitioning the ontology into smaller sections for more efficient processing.

5.5.1 Mapping Selection

Once the similarity scores have been calculated, we can take various approaches to identify the most valuable and reliable mapping between the two ontologies, for instance:

- **Absolute Threshold (AT)**: Select mappings with a similarity score above AT.
- **Relative Threshold (RT)**: Select mappings with a score greater than MaxScore - RT.
- **Rate (n%)**: Select the top n% of highest scores in set A.

For instance, let's consider the following concepts from two ontologies:

Concepts	**book**	**translator**	**editor**	**author**
product	0.84	0	0.9	0.12
provider	0.12	0	0.84	0.6
artist	0.6	0.05	0.12	0.84

Table 1: Similarity Scores between Concepts

If we select the following **Absolute Threshold**, **Relative Threshold** and **Rate**, then we obtain:

- **Absolute Threshold (0.7)**: Selects pairs with scores above 0.7: (product, book), (product, editor), (provider, editor), (artist, author).
- **Relative Threshold (delta 0.3)**: Selects pairs where scores are within 0.3 of the highest score (0.9). This includes: (product, book), (product, editor), (provider, editor), (provider, author), (artist, book), (artist, author).
- **Rate (30%)**: Selects the top 30% of pairs (4 pairs) with the highest scores: (product, book), (product, editor), (provider, editor), (artist, author).