Pontifícia Universidade Católica
do Rio de Janeiro

João Pedro Valladão Pinheiro

Improving the Quality of the User Experience by Query Answer Modification

Dissertação de Mestrado

Dissertation presented to the Programa de Pós–graduação em Informática of PUC-Rio in partial fulfillment of the requirements for the degree of Mestre em Informática.

Advisor : Prof. Marco Antonio Casanova
Co-advisor: Profa. Elisa Souza Menendez

Rio de Janeiro
April 2021

PUC-Rio - Certificação Digital Nº 1912709/CA

Pontifícia Universidade Católica
do Rio de Janeiro

João Pedro Valladão Pinheiro

Improving the Quality of the User Experience by Query Answer Modification

Dissertation presented to the Programa de Pós–graduação em Informática of PUC-Rio in partial fulfillment of the requirements for the degree of Mestre em Informática. Approved by the Examination Committee:

Prof. Marco Antonio Casanova
Advisor
Departamento de Informática – PUC-Rio

Profa. Elisa Souza Menendez
Co-Advisor
Campus Xique-Xique – Instituto Federal de Educação, Ciência e Tecnologia Baiano

Prof. Antonio Luz Furtado
Departamento de Informática – PUC-Rio

Prof. Luiz André Portes Paes Leme
Departamento de Ciências da Computação – UFF

Rio de Janeiro, April 30th, 2021

PUC-Rio - Certificação Digital Nº 1912709/CA

All rights reserved.

João Pedro Valladão Pinheiro

João Pedro Valladão Pinheiro holds a bachelor degree in Computer Engineering from Pontifical Catholic University of Rio de Janeiro (PUC-Rio). His main research topics are Semantic Web and Information Retrieval.

PUC-Rio - Certificação Digital Nº 1912709/CA

Bibliographic data

Pinheiro, João Pedro V.

Improving the Quality of the User Experience by Query Answer Modification / João Pedro Valladão Pinheiro; advisor: Marco Antonio Casanova; co-advisor: Elisa Souza Menendez. – 2021.

55 f: il. color. ; 30 cm

Dissertação (mestrado) - Pontifícia Universidade Católica do Rio de Janeiro, Departamento de Informática, 2021.

Inclui bibliografia

1. Computer Science – Teses. 2. Informatics – Teses. 3. Pergunta e Resposta (QA). 4. Consulta em Linguagem Natural (NLQ). 5. Agregação. 6. RDF. 7. Web Semântica . I. Casanova, Marco A.. II. Menendez, Elisa S.. III. Pontifícia Universidade Católica do Rio de Janeiro. Departamento de Informática. IV. Título.

CDD: 004

Acknowledgments

First, a special thank you to my advisor, Prof. Marco Antonio Casanova. For sure, his full support, patience, and wisdom were key contributors to my academic achievements. Prof. Casanova brilliantly guided me in my first article publication. Also, he was very comprehensive with the birth of my daughter. I will always remember this.

I would like to thank my co-advisor, Profa. Elisa Souza Menendez. Who could imagine that an email related to QALD dataset would result in an academic partnership? Profa. Elisa gave me excellent insights related to her deep experience with RDF Knowledge Bases. Really thanks for joining us on this journey.

I can't forget to thank all the staff, professors, and my classmates from the Department of Informatics of PUC-Rio, for the hours of work, study, and fun. Especially to Alexandre Novello, Claudio Escudero, Lauro de Souza and Luis Eduardo Craizer.

Last, but not least important, my deep gratitude to my wife, Amanda, for her love, support, partnership, and encouragement during these years. Our daughter, Maria Eduarda, reflects our love. Also, my gratitude to my parents, Maristela and Pedro Carlos, who always encourage me to invest in my education.

This study was financed in part by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior – Brasil (CAPES) – Finance Code 001.

PUC-Rio - Certificação Digital Nº 1912709/CA

Abstract

Pinheiro, João Pedro V.; Casanova, Marco A. (Advisor); Menendez, Elisa S. (Co-Advisor). **Improving the Quality of the User Experience by Query Answer Modification**. Rio de Janeiro, 2021. 55p. Dissertação de Mestrado – Departamento de Informática, Pontifícia Universidade Católica do Rio de Janeiro.

The answer of a query, submitted to a database or a knowledge base, is often long and may contain redundant data. The user is frequently forced to browse thru a long answer, or to refine and repeat the query until the answer reaches a manageable size. Without proper treatment, consuming the query answer may indeed become a tedious task. This study then proposes a process that modifies the presentation of a query answer to improve the quality of the user's experience, in the context of an RDF knowledge base. The process reorganizes the original query answer by applying heuristics to summarize the results. The original SPARQL query is modified and an exploration over the result set starts thru a guided navigation over predicates and its facets. The article also includes experiments based on RDF versions of MusicBrainz, enriched with DBpedia data, and IMDb, each with over 200 million RDF triples. The experiments use sample queries from well-known benchmarks.

PUC-Rio - Certificação Digital Nº 1912709/CA

Keywords

Question Answering (QA); Natural Language Query (NLQ); Aggregation; RDF; Semantic Web.

Resumo

Pinheiro, João Pedro V.; Casanova, Marco A.; Menendez, Elisa S.. **Melhorando a Qualidade da Experiência do Usuário através da Modificação da Resposta da Consulta**. Rio de Janeiro, 2021. 55p. Dissertação de Mestrado – Departamento de Informática, Pontifícia Universidade Católica do Rio de Janeiro.

A resposta de uma consulta, submetida a um banco de dados ou base de conhecimento, geralmente é longa e pode conter dados redundantes. O usuário é frequentemente forçado a navegar por uma longa resposta, ou refinar e repetir a consulta até que a resposta atinja um tamanho gerenciável. Sem o tratamento adequado, consumir a resposta da consulta pode se tornar uma tarefa tediosa. Este estudo, então, propõe um processo que modifica a apresentação da resposta da consulta para melhorar a qualidade de experiência do usuário, no contexto de uma base de conhecimento RDF. O processo reorganiza a resposta da consulta original aplicando heurísticas para comprimir os resultados. A consulta SPARQL original é modificada e uma exploração sobre o conjunto de resultados começa através de uma navegação guiada sobre predicados e suas facetas. O artigo também inclui experimentos baseados em versões RDF do MusicBrainz, enriquecido com dados do DBpedia, e IMDb, cada um com mais de 200 milhões de triplas RDF. Os experimentos utilizam exemplos de consultas de benchmarks conhecidos.

PUC-Rio - Certificação Digital Nº 1912709/CA

Palavras-chave

Pergunta e Resposta (QA); Consulta em Linguagem Natural (NLQ); Agregação; RDF; Web Semântica.

Table of contents

PUC-Rio - Certificação Digital Nº 1912709/CA

List of figures

PUC-Rio - Certificação Digital Nº 1912709/CA

List of tables

PUC-Rio - Certificação Digital Nº 1912709/CA

List of Abreviations

ISI – USC / Information Sciences Institute
ML – Machine Learning
NLP – Natural Language Processing
NLQ – Natural Language Query
OWL – Web Ontology Language
QA – Question Answering
RDF – Resource Description Framework
RDFS – Resource Description Framework Schema
SBBD – Brazilian Symposium on Databases
SBC – Brazilian Computer Society
SPARQL – SPARQL Protocol and RDF Query Language
W3C – World Wide Web Consortium

PUC-Rio - Certificação Digital Nº 1912709/CA

1
Introduction

Question Answering (QA) systems combine techniques from multiple fields of computer science, among which Natural Language Processing (NLP), Information Retrieval, Machine Learning (ML), and Semantic Web. Assuming that the user is interested in querying a database or a knowledge base, a QA system may be split into two parts: *question*, which receives a user's input in natural language, transforms it into a structured query and searches the data; and *answer*, which displays consistent results in a human-readable format to the user. The answer of a query is often long and may contain redundant data. The user is frequently forced to browse thru a long answer, or to refine and repeat the query until the answer reaches a manageable size. Without proper treatment, consuming the query answer may indeed become a tedious task.

This study addresses the problem of *query answer modification* to improve the quality of the user's experience, in the context of an RDF knowledge base. For example, imagine yourself as a user interacting with a voice virtual assistant, and you ask an open-ended question about a specific subject, e.g., *"Which artists were born on May 30th?"*. The query answer may have a long list of artists, partly shown in Table 1.1. Instead of listing the results, the virtual assistant may formulate questions to the user based on the prior result set, such as: *"Do you want to list American or European artists?"*; *"Do you prefer Jazz, Pop, or Classical music?"*; and *"Do you want to filter by active artists?"*.

Artist	Genre	Birth Date	Death Date	Gender	Nationality
Goodman, Benny	Jazz	1909-05-30	1986-06-13	Male	American
Leonhardt, Gustav	Classical	1928-05-30	2012-01-16	Male	Dutch
Green, CeeLo	Pop	1974-05-30		Male	American
Biosphere	Eletronic	1962-05-30		Male	Norwegian
Fredriksson, Marie	Pop	1958-05-30	2019-12-09	Female	Swedish
Banhart, Devendra	Folk	1981-05-30		Male	American

Table 1.1: Example of question answer for an open-ended question

Another common example, consider a user interacting with keyword search web systems. The goal is to find all movies starred by the actor Denzel Washington. Instead of typing the complete question "Which movies did Denzel Washington starred?", the user search only for the term "Denzel

PUC-Rio - Certificação Digital Nº 1912709/CA

Washington". The web system returns the main results centered and available filters aligned on the left position, as shown in Figure 1.1. The properties listed next to plus button (+) are called predicates. After clicking on the plus button, a drop-down list is displayed to the user. The options available in this drop-down list are called facets. Thus, the RDF graph exploration is done thru facet navigation.

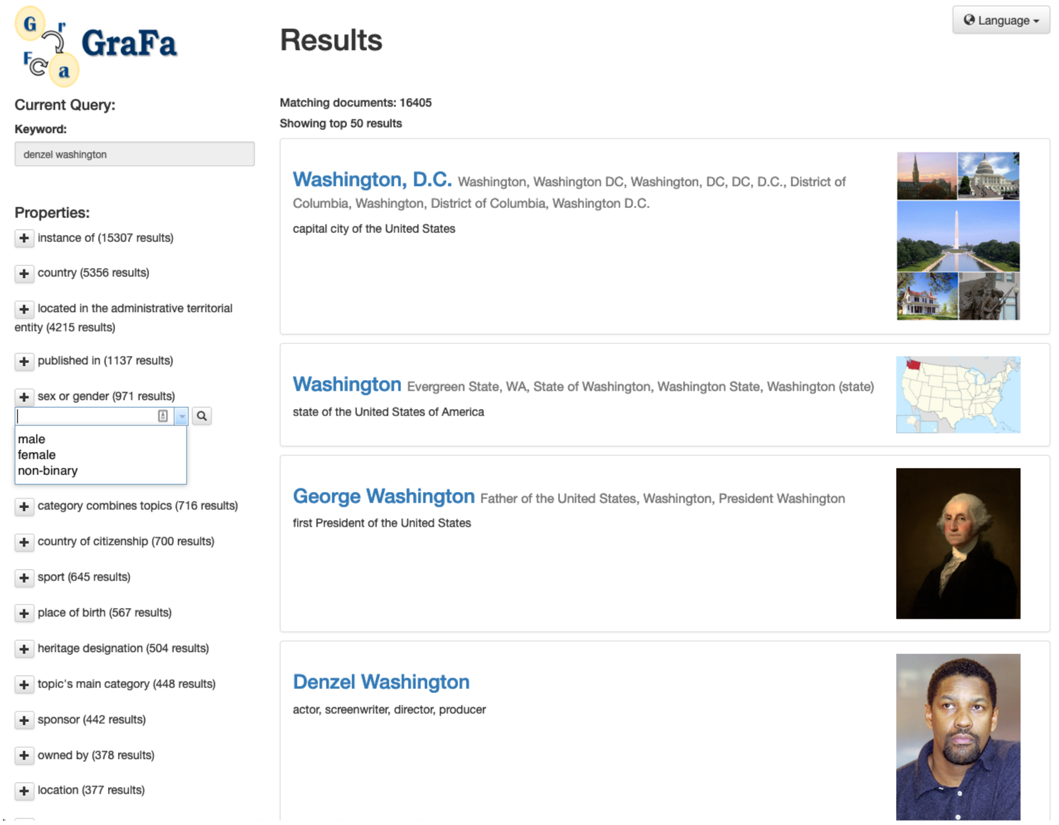


Figure 1.1: Search result for the input "Denzel Washington"

The dissertation proposes a fully automated process that reorganizes the original query answer by applying heuristics to summarize the results - explained in detail on Chapter 3. The original SPARQL query is modified and an exploration over the result set starts thru a guided navigation over predicates and its facets. The heuristics together with a set of thresholds allow deciding which properties returned in the query answer are interesting to apply aggregations (**group by** operations). Also, these definitions help the process to decide if the answer is ready to be displayed to the user, or if the answer must be improved. The decisions are based on global statistics about the RDF dataset, obtained a priori, and local statistics about the query answer, obtained dynamically. The statistics are related to the frequency of the class instances and the frequency of the predicates.

The dissertation also includes experiments based on the RDF versions of

PUC-Rio - Certificação Digital Nº 1912709/CA

MusicBrainz and IMDb described in [8]. The authors enriched a MusicBrainz dump with DBpedia data and transformed an IMDb relational database to RDF via R2RML. Each RDF dataset has over 200 million triples. The experiments use sample queries from the QALD - Question Answering over Linked Data[1] challenge and from Coffman's benchmark [2].

The rest of the dissertation is organized as follows. Chapter 2 introduces background concepts and summarizes related work. Chapter 3 discusses the query answer modification process. Chapter 4 describes the experiments and compare the results. Finally, Chapter 5 presents the conclusions and directions for future research.

PUC-Rio - Certificação Digital Nº 1912709/CA

[1] http://qald.aksw.org

2
Background and Related Work

In this chapter we introduce important background concepts and present related work. Section 2.1.1 describes key aspects of Linked Data and RDF structure. Section 2.1.2 explains communication between humans and machines. Section 2.2 presents the related work about aggregation and faceted search approaches. Finally, Section 2.3 highlights the key points presented in this chapter.

2.1
Background

2.1.1
Linked Data

In [1], the term Semantic Web emerged as an extension from the classic World Wide Web. The key concept was to provide meaningful contents of Web pages that were machine readable. Thus, a collection of Semantic Web technologies was developed and recommended by W3C. These technologies compose an environment where data is connected, new connections can be inferred, data can be queried, and many other features. This Web of interrelated data can also be referred to as Linked Data.

As part of W3C, RDF is a standard model for data interchange on the web. A key characteristic of RDF is the presence of data and metadata combined which makes it flexible to support the evolution of schemes over time without breaking the way data is consumed.

An RDF Triple is the basic unit of data stored in RDF. The linking structure forms a directed, labeled graph, where the edges represent the named link between two resources, represented by graph nodes - (subject, predicate, object). This simple structure is very powerful enabling application in multiple scenarios.

For instance, Figure 2.1 represents the sentence *"Paul Schuster was born in Dresden"*. The graph representation is from Wikidata[1] which is described using schema.org[2] vocabulary. The black arrows represent the sentence itself.

[1]https://www.wikidata.org/wiki/Wikidata:Main_Page
[2]https://schema.org/docs/about.html

PUC-Rio - Certificação Digital Nº 1912709/CA

The blue and green arrows provide enriched information. And the red dashed arrow is an inferred relationship.

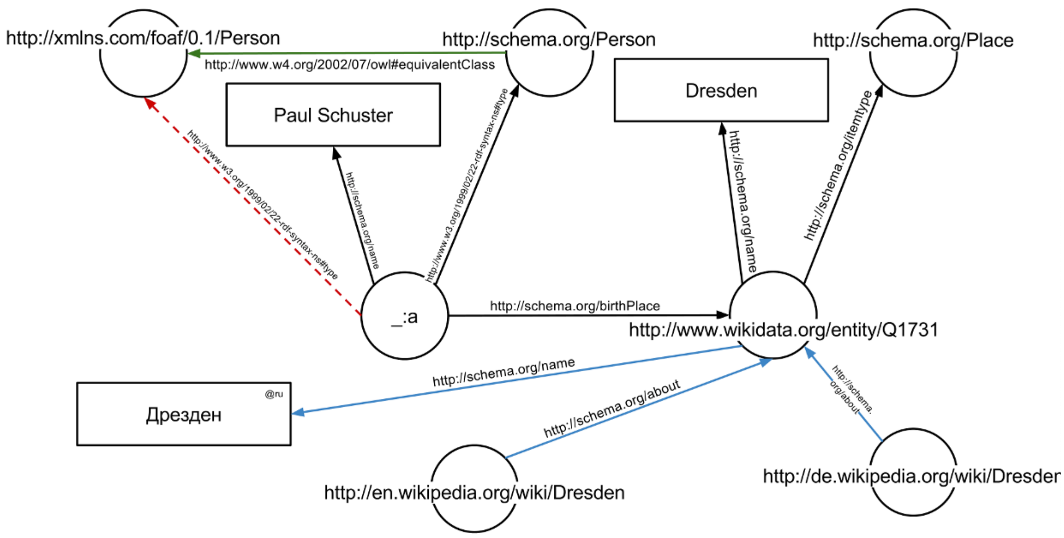


Figure 2.1: RDF example

RDF is used in combination with vocabularies that provide semantic information about the resources. Also part of W3C, OWL is an example of vocabulary. It is a formal language which allows more expressiveness for RDFS. Once again in Figure 2.1, the green arrow denotes the OWL's primitive *equivalent class*. It connects classes from two different vocabularies - `http://schema.org/Person` and `http://xmlns.com/foaf/0.1/Person`. Only because of this connection it is possible to infer the red dashed relationship.

Finally, another important task related to Linked Data is information retrieval. W3C defined a query language for RDF graphs called SPARQL. A simple example of SPARQL query is shown below, which retrieves the URIs for all movies performed by *"Denzel Washington"*.

```
prefix quira: <http://www.quira.org/>

select distinct ?movie
where {
  ?movie a quira:Movie .
  ?movie quira:actor ?actor .
  ?actor quira:name ``Denzel Washington'' .
}
```

The *query form* `select` followed by the *solution modifier* `distinct` guarantees that only unique URI will be presented as result. The `where` *clause*

PUC-Rio - Certificação Digital Nº 1912709/CA

restricts the result applying a graph pattern matching over the RDF graph. Also the `a` is a sugar syntax for the predicate `rdf:type`. Important to mention that SPARQL also supports aggregation and subqueries, which are the main topics of this dissertation. These topics are described in Section 3.

2.1.2 Question Answering

In a seminal paper, Webber proposed a theoretical framework that divided the communication between humans and machines into three parts [11]. Instead of only considering questions and answers, Webber proposed a clear distinction between a question, an answer and a response. A question is a request by a user demanding information or to perform an action. An answer is the information or performance directly requested. A response embraces multiple elements, such as direct answer, additional information or actions that fulfill the answer, information or actions related to the original request instead of an answer, and suggested additional information or actions since there is no proper answer (also called "did you mean?").

Comparing the proposed framework with our work, we consider only open-ended questions. Thus, we can assume that a question is strictly demanding information. From the point of view of the answer (called "response" in the framework), our process suggests information related to the original request instead of a single answer. This information is presented to the user as facets allowing a dialog with the user and guiding him to the desired answer.

PUC-Rio - Certificação Digital Nº 1912709/CA

2.2 Related Work

2.2.1 Aggregation and Summarization

Knowledge Base systems are usually constructed from multiple sources, which may lead to the generation of duplicated data. By contrast, humans avoid redundancy in the act of writing or speaking. Indeed, reducing duplicated data in the communication between humans and machines is a challenging task. Aggregation and summarization are important techniques that help solving this issue.

The problem of redundancy is addressed, for example, in [3]. The authors suggest aggregation strategies to remove redundancy from text - usually retrieved answers from databases. An interesting example, used in this paper, to illustrate the problem goes as follows. Consider the question:

```
``Who is currently at ISI?''
```

Suppose that the answer to this question is:

```
``Yigal is an employee at ISI. Hercules is a visitor at ISI.
Eduard is an employee at ISI. Kevin is an employee at ISI.
Vibhu is a student at ISI.''
```

Note that the answer is too long and repetitive. After applying the aggregation rules suggested in the paper, the modified answer is shortened and easier to understand:

```
``At ISI, Yigal, Eduard, and Kevin are all employees;
Vibhu is a student; and Hercules is a visitor.''
```

In [4], the main idea was an approach to present query results as sentences in Natural Language (NL) with provenance information. The authors argued that the answers in the query result lack justification and suggested the notion of provenance, which corresponds to including additional information to query results. Also, provenance information helps validate answers. The proposed solution consisted of the following key contributions: provenance tracking based on the NL query structure, factorization, summarization, and implementation and experiments.

PUC-Rio - Certificação Digital Nº 1912709/CA

2.2.2 Faceted Browsing

In a similar direction, faceted browsing [10] (also called "faceted search" [12], as in Figure 1.1) is a complement to keyword search which provides an iterative way to refine search results. Facets are usually displayed to the user as rectangles right next to the main list results provided by keyword search. These facets contain relevant grouped information which guide users to the desired answer.

In [9] and [7], faceted browsing is used to simplify the user's interaction with data. The first reference [9] allows search by keyword or type. A type would be an IRI from the RDF graph. While there are facets available, the user can navigate interacting only with them. It is important to mention that facets with zero results are never offered. The second paper [7] uses faceted browsing for user evaluation purpose. With fixed *subject* and *predicate*, the user receives a list of *objects* sorted by the so-called *TripleRank* and decides if each of them is related, don't know, or not related.

2.3 Chapter Conclusion

In summary, several studies addressed the problem of creating a question-answering (QA) interface to databases. Usually, the proposed QA process has four steps: *Question Analysis*, *Phrase Mapping*, *Disambiguation*, and *Query Construction* - not necessarily in this order [5]. In this dissertation, we assume that the QA interface is constructed over an RDF knowledge base, accessed through a SPARQL endpoint.

PUC-Rio - Certificação Digital Nº 1912709/CA

3
The query answer modification process

This chapter is organized as follows. Section 3.1 explains the basic definitions using heuristics and thresholds. Section 3.2 describes the process of transforming a single-column into a three-column result set. Section 3.3 addresses the use of frequency analysis based on RDF metadata. Section 3.4 discusses how the stop condition is applied. Finally, Section 3.5 highlights the key points presented in this chapter.

3.1
Heuristics and Thresholds

During the research about topics related to this dissertation, we noticed that we could approach the main topic in many ways. We preferred to test a set of heuristics together with parameterized thresholds. This decision made the project simple but very powerful.

The heuristics tested are related to user's navigation thru data. The goal was to automate the predicate/facet decision, proposing promising paths, while thresholds help cut undesirable branches, reduce result set and control the stop condition.

The heuristics tested were:

– Σ : from the most **embracing** predicate, select the most **embracing** facet

– Π : from the most **restrictive** predicate, select the most **embracing** facet

– Ω : select the most **embracing** facet, regardless the predicate

In this context, a predicate or facet is restrictive if its selection reduce the total number of rows of the result set. On the other hand, an embracing predicate or facet means its selection keep or barely reduce the total number of rows of the result set. Recall that facets are usually displayed right next to the keyword search results and contain relevant grouped information which helps users to navigate and find the desired answer. In Table 3.1, there are examples of these two cases.

PUC-Rio - Certificação Digital Nº 1912709/CA

Initial Result Set	49 rows
Predicate/Facet selected	`imdb:label` => `Columbia/Tristar`
Final Result Set	5 rows
Compression Rate	89,80%

(a) Restrictive predicate/facet selection

Initial Result Set	91 rows
Predicate/Facet selected	`imdb:color_info` => `Color`
Final Result Set	74 rows
Compression Rate	18,68%

(b) Embracing predicate/facet selection

Table 3.1: Examples of restrictive and embracing predicates/facets

The decision about selecting the most embracing facet was empirical. Beyond that, returning the most restrictive facet means, in most cases, display a single result to the user. A single result returned guarantees an excellent compression rate, but it may reduce accuracy. On Chapter 4, a deeper discussion over decisions and results can be found.

And here are the thresholds defined:

- α : max number of predicates' distinct values
- β : max number of unique subjects to be returned to the user
- δ : min and max range of predicates' rate presence over unique subjects

For better understanding, it is important to mention that the examples on this chapter assume the use of the Σ heuristic. Also, to simplify the visualization, we decided to illustrate the example flows as K-D trees. On Chapter 4, we perform multiple use cases to analyse the results of each proposed heuristic and describe each dataset mentioned here - IMDb and MusicBrainz.

As the first example, consider the question "Give me all movies by Denzel Washington" over IMDb dataset.

On Figure 3.1, we represent the question as the root node. The following nodes have two dimensions (K-D tree with k = 2). The first dimension is the predicate, and the second is the facets. Some predicates are collapsed because the number of facets are greater than α. The number in parenthesis right next the predicate represents the number of facets. There are three types of arrows: (I) gray which represents the ignored predicates; (II) solid black which represents selectable predicates; (III) bold solid black which represents the selected predicate.

PUC-Rio - Certificação Digital Nº 1912709/CA

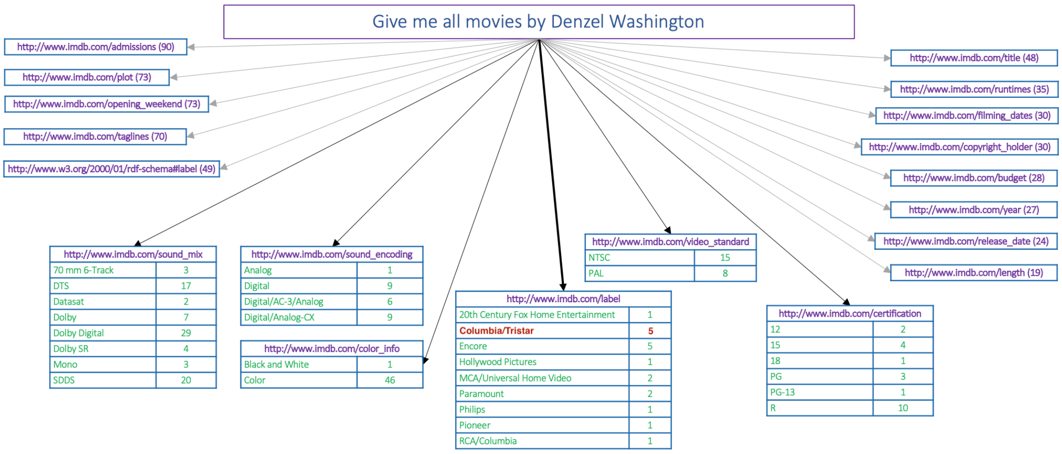


Figure 3.1: K-D tree structure representing the only step over IMDb dataset

Thus, the proposed algorithm selects the facet `Columbia/Tristar` from the predicate `http://www.imdb.com/label`. In this case, the algorithm also suggests to stop navigation and presents the final result set. This decision is related to β threshold - the number of distinct movies was less or equal than β.

As the second example, let's use the question "Which artists were born on May 30th?" over MusicBrainz dataset.

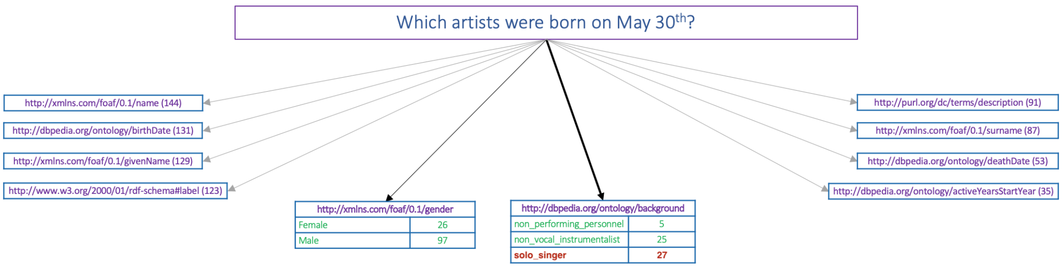


Figure 3.2: K-D tree structure representing 1st step over MusicBrainz dataset

Figures 3.2 and 3.3 represent the interactions necessary until an appropriate result set is found. The main difference between the two examples are the number of interactions. While the first example finds a solution in a single step, the second example needs two interactions to find a solution. There are no restrictions in the number of steps given until the final answer.

In this case, the first step selects the facet `solo_singer` from the predicate `http://dbpedia.org/ontology/background` and the second step selects the facet `Singer-songwriter, musician` from the predicate `http://dbpedia.org/property/occupation`.

Notice that the predicate `http://dbpedia.org/property/occupation` was not an option on Figure 3.2. This behavior is expected and related to α or

PUC-Rio - Certificação Digital Nº 1912709/CA

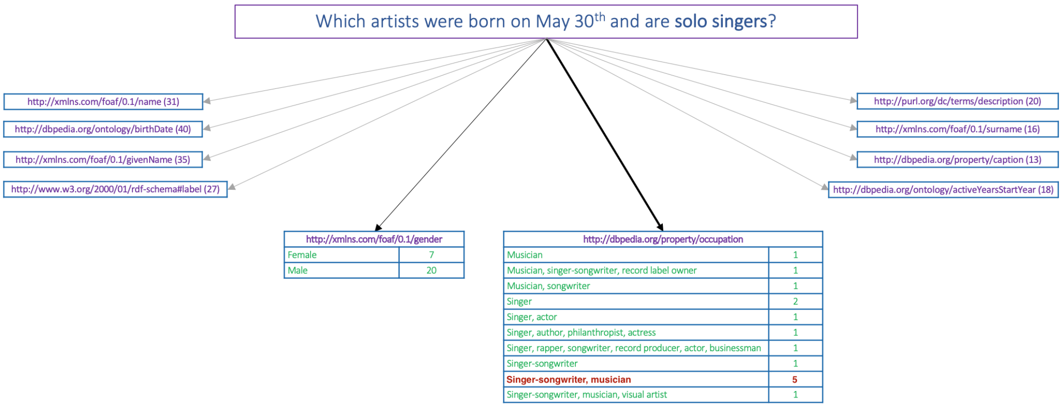


Figure 3.3: K-D tree structure representing 2nd step over MusicBrainz dataset

δ range thresholds. Since the state is recalculated at each step, some discarded predicates first may be selectable on next steps.

3.2 Transforming single-column into three-column result sets

The query answer modification process we propose starts after the query is executed. The expected input is the SPARQL query as illustrated in Figure 3.4. There are two possible scenarios: the result set has a *single column*, or the result set has *multiple columns*. Our study focuses on the first case.

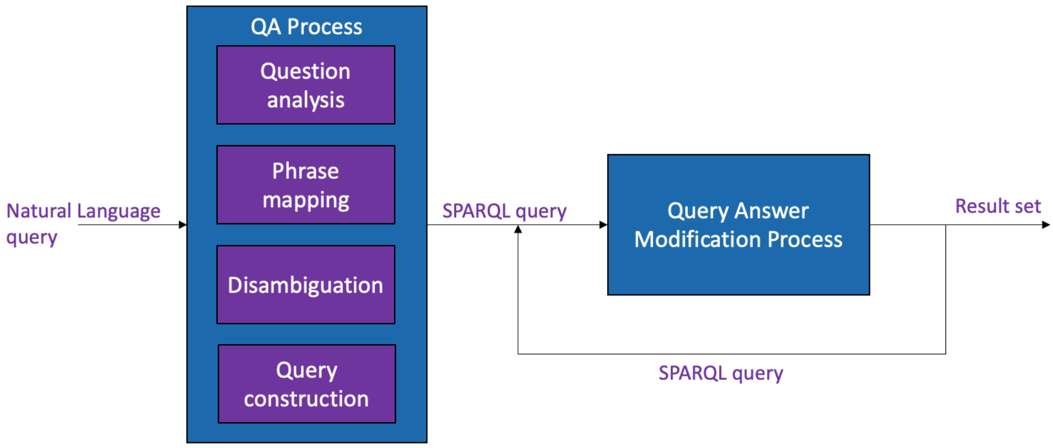


Figure 3.4: Query answer modification process

We base our discussion on a series of question answering challenges over Linked Data, referred to as QALD - Question Answering over Linked Data[1]. Several papers use QALD to measure quality metrics of system's answers [6]. We noticed that most queries listed in the QALD challenges had single-column answers, which calls for enriching the answers for the purposes of this paper. A simple approach is to add to the instances returned their property

[1]http://qald.aksw.org

PUC-Rio - Certificação Digital Nº 1912709/CA

values. Indeed, frequently, the answers represent sets of instances of the same `rdf:type`. So, it is straightforward to modify the original SPARQL query to also retrieve the desired property values.

As an example, consider the question *"Which artists were born on May 30th?"*. The result set of the corresponding SPARQL query has instances of type `mo:MusicArtist`, as in Figure 3.5(c). Then, by modifying the original SPARQL query, it is possible to also retrieve property values, as shown in Figure 3.5(d). Note that, in Figure 3.5(d), the column *artist* has repeated values. However, instead of normalizing the returned table, we decided to keep this three-column format to simplify data manipulation.

```
prefix mo: <http://purl.org/ontology/mo/>
prefix dbo: <http://dbpedia.org/ontology/>

select distinct ?artist
where {
  ?artist a mo:MusicArtist .
  ?artist dbo:birthDate ?date .
  filter(regex(?date, "5-30$", "i")) .
}
```

(a) Single-column query

```
select distinct ?artist ?predicate ?object
where {
  {
    select ?artist
    where {
      ?artist a mo:MusicArtist .
      ?artist dbo:birthDate ?date .
      filter(regex(?date, "5-30$", "i"))
    }
  } # graph pattern 1 - prior query
  . # conjunction (inner join)
  {
    select ?artist ?predicate ?object
    where {
      ?artist ?predicate ?object .
      filter(isLiteral(?object))
    }
  } # graph pattern 2
}
```

(b) Three-column query

artist
mo:MusicArtist/1
mo:MusicArtist/2
mo:MusicArtist/3
mo:MusicArtist/4
mo:MusicArtist/5
mo:MusicArtist/6
mo:MusicArtist/7
mo:MusicArtist/8
mo:MusicArtist/9
mo:MusicArtist/10

(c) Single-column result set

artist	predicate	object
mo:MusicArtist/1	foaf:name	"Green, CeeLo"
mo:MusicArtist/1	mo:genre	"pop"
mo:MusicArtist/1	dbo:BirthDate	"1974-05-30"
mo:MusicArtist/1	dbo:DeathDate	""
mo:MusicArtist/1	foaf:gender	"Male"
mo:MusicArtist/1	dbp:nationality	"American"
mo:MusicArtist/2	foaf:name	"Leonhardt, Gustav"
mo:MusicArtist/2	mo:genre	"Classical"
mo:MusicArtist/2	dbo:BirthDate	"1928-05-30"
mo:MusicArtist/2	dbo:DeathDate	"2012-01-16"
mo:MusicArtist/2	foaf:gender	"Male"
mo:MusicArtist/2	dbp:nationality	"Dutch"
mo:MusicArtist/3	foaf:name	"Goodman, Benny"
mo:MusicArtist/3	mo:genre	"Jazz"
mo:MusicArtist/3	dbo:BirthDate	"1909-05-30"
mo:MusicArtist/3	dbo:DeathDate	"1986-06-13"
mo:MusicArtist/3	foaf:gender	"Male"
mo:MusicArtist/3	dbp:nationality	"American"

(d) Three-column result set

Figure 3.5: Transformation with SPARQL queries

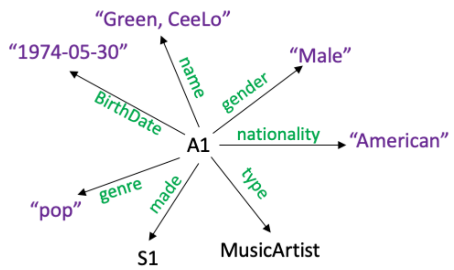
PUC-Rio - Certificação Digital Nº 1912709/CA

3.3
Frequency analysis based on computed metadata

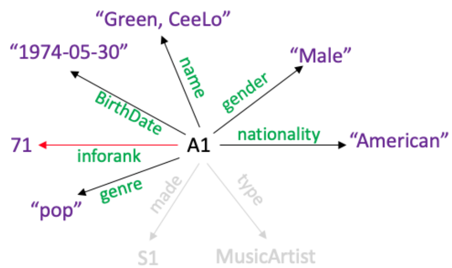
In the process we propose, a set of SPARQL queries is used to generate statistics of the RDF graph, which help decide what to do next. These statistics are related to the frequency of the instances by class and the frequency of the predicates.

There are two types of frequencies used. A *global frequency* is defined over the full graph and is computed only once before any query is executed. On the other hand, a *local frequency* is defined over the sub-graph generated as in Section 3.2 and is computed at run time. It is important to highlight that both *global* and *local frequencies* are computed over predicates pointing to literals only.

Entity ranking is based on *InfoRank*, a family of importance measures proposed in [8]. The proposed importance measures are combinations of three intuitions: (I) "important things have lots of information about them"; (II) "important things are surrounded by other important things"; (III) "few important relations (e.g. friends) are better than many unimportant relations (e.g. acquaintances)". Hence, the strategy is based on the level of informativeness of an entity, represented as literals in RDF graphs. They use a PageRank inspired approach to propagate the importance scores from entity to entity. The *InfoRank* metric helps our process prioritize the most relevant triples of the result set.



(a) All predicates



(b) Filtered predicates

Figure 3.6: Filtered predicates by literal and highlighted *InfoRank*

As an example, Figure 3.6 shows an instance **A1**. The initial state (Figure 3.6(a)) has predicates pointing to literals and other instances. Notice that the final state (Figure 3.6(b)) only has predicates pointing to literals and an extra predicate called *inforank*.

Parameterized thresholds are used to filter predicates that are candidates to be used in a **group_by** operation. By default, these threshold values are set between δ_{min} and δ_{max}, which means the predicate must appear in at least $(\delta_{min} * 100)\%$ and not more than $(\delta_{max} * 100)\%$ of the unique subjects. For

PUC-Rio - Certificação Digital Nº 1912709/CA

clarity, consider again the question *"Which artists were born on May 30th?"* and values $\delta_{min} = 0.4; \delta_{max} = 2.0$. Analysing the three-column result set, the predicate `rdfs:comment` appears 497 times and there are 123 unique artists. This means the predicate appears 4.04/artist on the average. Thus, the predicate `rdfs:comment` is removed by the process because it exceeds δ_{max}.

3.4 Verifying stop condition

As mentioned in the previous section, an aggregation process is applied over the filtered predicates. These predicates are evaluated and sorted by its *local frequency*. After applying the filter related to the α threshold, the process automatically chooses the facet and verifies if another interaction is necessary, or if the result set is ready to be displayed to the user.

For clarity, consider once again the question *"Which artists were born on May 30th?"* and value $\alpha = 10$. Analysing the previous filtered candidates, the predicate `dbo:activeYearsStartYear` has 35 unique values. Although it might be a promising candidate, the predicate `dbo:activeYearsStartYear` is removed by the process because it exceeds α.

Using the same example, the filtered predicates are: (I) `foaf:gender`, with two aggregated values - `female` and `male`; (II) `dbo:background`, with three aggregated values - `non_performing_personnel`, `non_vocal_instrumentalist`, and `solo_singer`.

Since `dbo:backgroud` precedes `foaf:gender` in the computed frequencies, a facet related to `dbo:backgroud` is chosen. As `solo_singer` is the most embracing facet, the process automatically chooses it and verifies the stop condition.

If the number of unique subjects is greater than the threshold β, the whole process restarts. Otherwise, the process finishes and the final result set is presented to the user. The process also finishes if there is an empty set of selectable predicates.

3.5 Chapter Conclusion

The detailed steps of the query answer modification process described in the above sections are easily pluggable to any QA system over RDF. In fact, the presence of data and metadata combined allows the RDF graphs to be not schema oriented. Hence, it is possible to retrieve information of any RDF structure with the same SPARQL queries. Sections 3.3 and 3.2 take full advantage of this.

PUC-Rio - Certificação Digital Nº 1912709/CA

Moreover, the presented heuristics and parameterized thresholds have simplicity as main characteristic. The predicate filtering, facet selection, and stop condition verification are full responsibility of them.

PUC-Rio - Certificação Digital Nº 1912709/CA

4
Experiments

4.1
Setup

We performed initial experiments using the RDF versions of MusicBrainz and IMDb[1] described in [8]. The authors enriched a MusicBrainz dump with DBpedia data and transformed an IMDb relational database to RDF via R2RML. These datasets also contains the *InfoRank* scores of instances, properties and classes. Figures 4.1 and 4.2 show the resulting MusicBrainz and IMDb schemas, respectively. Each RDF dataset has over 200 million triples. We used sample queries from the QALD[2] challenge and Coffman's benchmark [2].

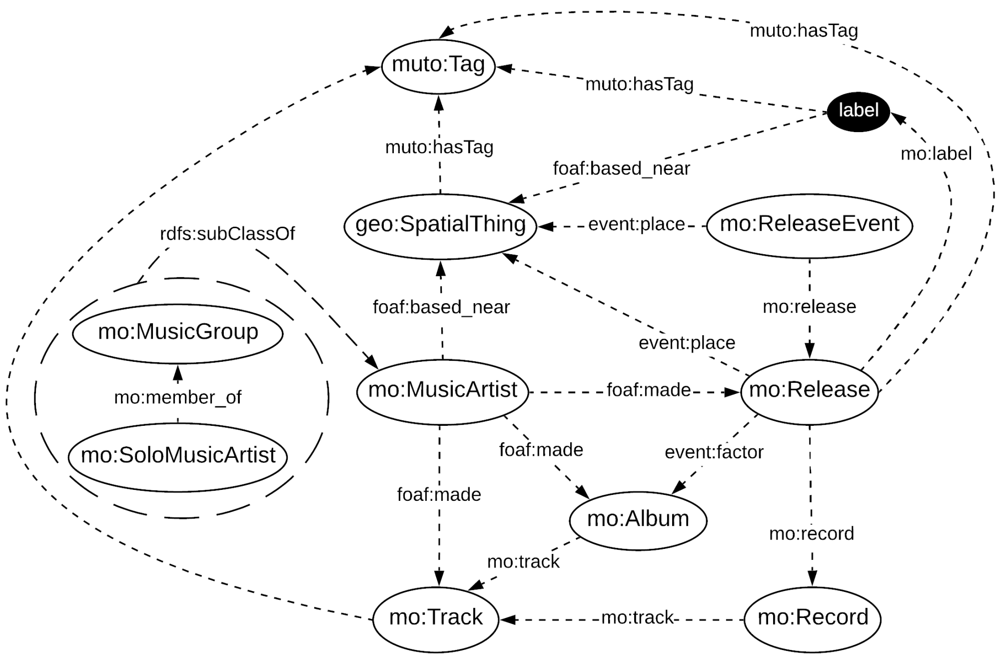


Figure 4.1: MusicBrainz Schema

[1]https://sites.google.com/view/quira/
[2]http://qald.aksw.org

PUC-Rio - Certificação Digital Nº 1912709/CA

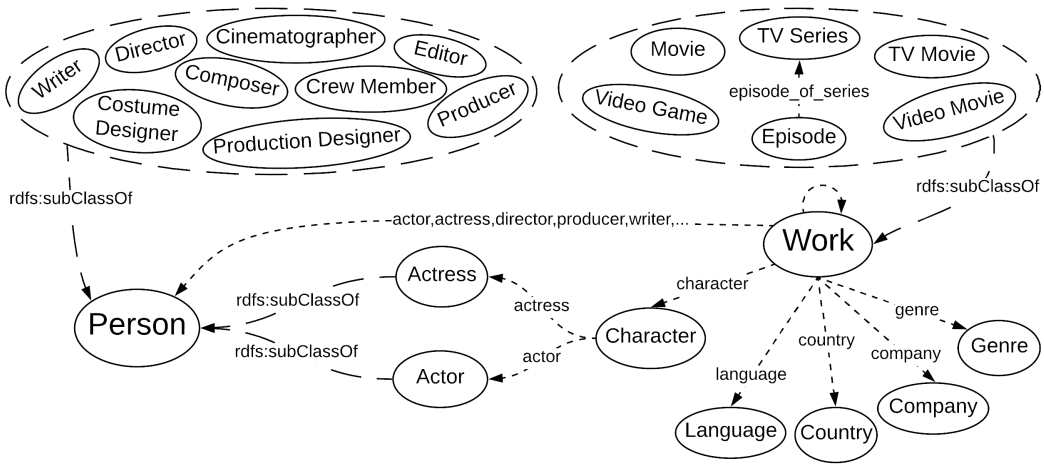


Figure 4.2: IMDb Schema

To store and manage the RDF datasets, we used the component TDB2 of Apache Jena for RDF[3]. Apache Jena Fuseki (a SPARQL server) ran on a server machine with OS GNU/Linux Ubuntu 16.04.6 LTS, a quad-core processor Intel(R) Core(TM) i7-5820K CPU @ 3.30GHz, 64 GB of RAM and SSD 1TB.

In the following sections, we discuss the effect of the heuristics proposed in section 3.1 over both datasets on the query result. The thresholds used for these experiments were: $\alpha = 10$, $\beta = 15$ and $\delta = (\delta_{min}, \delta_{max}) = (0.4, 2.0)$. Also, we refer to the proposed query reformulation process simply as the *process*, for brevity.

4.2 MusicBrainz Results

4.2.1 Overview

In this section, we detail the experiments with the QALD query for MusicBrainz presented earlier: *"Which artists were born on May 30th?"*. The initial process generated the SPARQL query below, with results ranked by the *InfoRank* score.

[3]https://jena.apache.org

PUC-Rio - Certificação Digital Nº 1912709/CA

```
prefix mo: <http://purl.org/ontology/mo/>
prefix dbo: <http://dbpedia.org/ontology/>
prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#>
prefix quira: <http://www.quira.org/>

select distinct ?artist ?label ?inforank
where {
  ?artist a mo:MusicArtist .
  ?artist dbo:birthDate ?date .
  ?artist rdfs:label ?label .
  ?artist quira:inforank ?inforank .
  filter (regex(?date, ``5-30\$'', ``i'')) .
}
order by desc(?inforank)
```

Table 4.1 shows a preview of the original result. Note that the SPARQL query returned 122 artists that were born on May 30th, which the user might consider to be a long list to interact with.

#	**Artist partial URI**	**Artist name**
1	/artist/b09ae88f-4156-4caa-b129-1cacb5e1632e	Benny Goodman
2	/artist/27b0750a-7318-4075-9470-43b82d454ea0	Gustav Leonhardt
3	/artist/2c69465c-0f76-45ce-90a2-1ed0fdacc997	CeeLo Green
...	...	...
120	/artist/e00871b0-f6b5-41cf-b758-f2f1ea467818	Frank St. Leger
121	/artist/09ffe9f4-d54e-4943-8297-4456963f0def	Josephine Preston Peabody
122	/artist/22b95e86-0749-4df1-ae29-cd5acfe5a285	Jim Murray

Table 4.1: Preview of the original SPARQL result

The β threshold is used to indicate when the (reformulation) process should be applied. For these experiments, we chose a maximum of 15 lines (β = 15), that is, a list with only 15 artists in this example. So, the stop condition is not valid and the process must go on. In the following subsections, we will be able to compare the three proposed heuristics.

PUC-Rio - Certificação Digital Nº 1912709/CA

4.2.2
Applying the Σ heuristic over MusicBrainz

As the stop condition was invalid in the previous step, the process reformulated the SPARQL query to capture the predicates that are candidates to be used for aggregation. The code on Listing 4.1 represents the original one-column result set transformed into three-column, mentioned on Section 3.2.

```
prefix mo: <http://purl.org/ontology/mo/>
prefix dbo: <http://dbpedia.org/ontology/>

select distinct ?artist ?predicate ?object
where {
  {
    select ?artist
    where {
      ?artist a mo:MusicArtist .
      ?artist dbo:birthDate ?date .
      filter (regex(?date, ``5-30\$'', ``i''))
    }
  }
  .
  {
    select ?artist ?predicate ?object
    where {
      ?artist ?predicate ?object .
      filter(isLiteral(?object))
    }
  }
}
```

Listing 4.1: Transforming single-column into three-column result set

The result was stored in memory to facilitate manipulation and to avoid further accesses to the database. Then, the process grouped the results by predicate and counted the distinct object values. Table 4.2 presents these results.

The α threshold is used to limit the maximum number of distinct values for the predicates, so the facets available to the user are not too long. Also, the δ threshold is used as a range of presence for the predicates. In Table 4.2, columns *Distinct Values* and *Appears* represent thresholds α and δ, respectively.

In the Σ heuristic the approach is to choose, from the set of predicates with less distinct values than the maximum, the predicate with the highest number of distinct values. Following this heuristic, the process chose the predicate `http://dbpedia.org/ontology/background`, which refers to the type of music artists.

PUC-Rio - Certificação Digital Nº 1912709/CA

#	Predicate	Appears	Distinct Values
1	http://dbpedia.org/ontology/birthDate	367	131
2	http://xmlns.com/foaf/0.1/givenName	247	129
3	http://xmlns.com/foaf/0.1/name	241	144
4	http://www.w3.org/2000/01/rdf-schema#label	203	123
5	http://xmlns.com/foaf/0.1/gender	203	2
6	http://purl.org/dc/terms/description	197	91
7	http://xmlns.com/foaf/0.1/surname	146	87
8	http://dbpedia.org/ontology/activeYearsStartYear	120	35
9	http://dbpedia.org/ontology/background	110	3
10	http://dbpedia.org/ontology/deathDate	84	53

Table 4.2: Available predicates in the 1^{st} reformulation process

Hence, the process now chooses the most embracing facet from the options presented in Table 4.3(a), which corresponds to the 3 distinct values of the predicate `http://dbpedia.org/ontology/background`. Following the Σ heuristic, the process chose facet `1. Solo singer`, which has one of the highest artist counts. The final result decreased to 27 artists, as shown in Table 4.3(b).

#	Options	Counts
1	Solo singer	27
2	Non vocal instrumentalist	25
3	Non-performing personnel	5

(a) Available facets for predicate artist background

#	Artist partial URI	Artist name
1	/artist/2c69465c-0f76-45ce-90a2-1ed0fdacc997	CeeLo Green
2	/artist/0110e63e-0a9b-4818-af8e-41e180c20b9a	Devendra Banhart
3	/artist/3a0373c0-f9c1-4eb3-9c10-53cc18193b07	Marie Fredriksson
...	...	...
25	/artist/0de740a2-a651-4d76-9cd5-54912a64070f	Gladys Horton
26	/artist/be0c5489-92e2-4149-b094-48293606f34b	Brian Fair
27	/artist/ae148627-23cc-48d3-a1a7-804f2af6b7dc	Rick DePiro (Ricky Dee)

(b) Preview of the result filtered by “Solo singer” background

Table 4.3: Results from 1^{st} reformulation process

Since this result was still higher than our β threshold, the process was reapplied. Again, the process grouped the results by predicate and counted the distinct object values. Table 4.4 presents this result.

Continuing with the proposed Σ heuristic, the process chose the predicate `http://dbpedia.org/property/occupation`, and a list with 10 available options to choose from was generated, as shown in Table 4.5(a).

Then, the process chose option `1. Singer-songwriter, musician` as the artist occupation. The process finally stopped, since it achieved our β

PUC-Rio - Certificação Digital Nº 1912709/CA

#	Predicate	Distinct Values
1	http://dbpedia.org/ontology/birthDate	40
2	http://xmlns.com/foaf/0.1/givenName	35
3	http://xmlns.com/foaf/0.1/name	31
4	http://www.w3.org/2000/01/rdf-schema#label	27
5	http://dbpedia.org/ontology/wikiPageID	26
6	http://dbpedia.org/ontology/wikiPageRevisionID	26
7	http://purl.org/dc/terms/description	20
8	http://dbpedia.org/ontology/activeYearsStartYear	18
9	http://xmlns.com/foaf/0.1/surname	16
10	http://dbpedia.org/property/caption	13
11	http://dbpedia.org/property/occupation	10
12	http://xmlns.com/foaf/0.1/gender	2

Table 4.4: Available predicates in the 2^{nd} reformulation process

threshold of 15 lines. Hence, the final result was presented to the user in a decreased order of *InfoRank* score, as shown in Table 4.5(b).

#	Options	Counts
1	Singer-songwriter, musician	5
2	Singer	2
3	Musician	1
4	Singer, actor	1
5	Musician, songwriter	1
6	Singer-songwriter	1
7	Singer, author, philanthropist, actress	1
8	Musician, singer-songwriter, record label owner	1
9	Singer-songwriter, musician, visual artist	1
10	Singer, rapper, songwriter, record producer, actor, businessman	1

(a) Available facets for predicate artist occupation

#	Artist partial URI	Artist name
1	/artist/a0580131-73f3-49c8-aac5-2c478f64a363	Stephen Duffy
2	/artist/19e07fd0-5642-47a0-a2b9-b8176e6b06e5	Brooke Waggoner
3	/artist/23c738ed-5dc4-4ff7-8c00-3c1c54e8eb89	Kevin Barnes
4	/artist/1d566a14-4094-4f96-abb7-969b4f439728	Geva Alon
5	/artist/4e0e884d-099b-4ca9-bf4d-bcb31e739540	Duffy

(b) The final result presented to the user

Table 4.5: Results from 2^{nd} reformulation process

PUC-Rio - Certificação Digital Nº 1912709/CA

4.2.3 Applying the Π heuristic over MusicBrainz

Since the original question resulted on a long result set, the process starts the 1^{st} interaction step. The result set related to the SPARQL query on Listing 4.1 is reused to capture the predicates that are candidates to be used for aggregation.

Basically, the three-column result set is grouped by *predicate*, and the metrics: *distinct values* and *presence* are calculated. These two metrics are related to previously defined thresholds α and δ, respectively. Table 4.6 presents the results.

#	Predicate	Appears	Distinct Values
1	http://dbpedia.org/ontology/birthDate	367	131
2	http://xmlns.com/foaf/0.1/givenName	247	129
3	http://xmlns.com/foaf/0.1/name	241	144
4	http://www.w3.org/2000/01/rdf-schema#label	203	123
5	http://xmlns.com/foaf/0.1/gender	203	2
6	http://purl.org/dc/terms/description	197	91
7	http://xmlns.com/foaf/0.1/surname	146	87
8	http://dbpedia.org/ontology/activeYearsStartYear	120	35
9	http://dbpedia.org/ontology/background	110	3
10	http://dbpedia.org/ontology/deathDate	84	53

Table 4.6: Available predicates in the 1^{st} reformulation process

Recall the Π heuristic, which the approach is to select the most **embracing** facet from the most **restrictive** predicate. In the Table 4.6, the most restrictive predicate is `http://xmlns.com/foaf/0.1/gender`. And the most embracing facet is `male`, as shown in Table 4.7(a). After the process chose option `1. male`, the final result decreased to 97 artists, as shown in Table 4.7(b).

#	Options	Counts
1	male	97
2	female	26

(a) Available facets for predicate artist gender

#	Artist partial URI	Artist name
1	/artist/b09ae88f-4156-4caa-b129-1cacb5e1632e	Benny Goodman
2	/artist/27b0750a-7318-4075-9470-43b82d454ea0	Gustav Leonhardt
3	/artist/2c69465c-0f76-45ce-90a2-1ed0fdacc997	CeeLo Green
...	...	...
95	/artist/5d0b474d-d24a-4b5c-8a67-7a79f7f9949a	Judd Woldin
96	/artist/e00871b0-f6b5-41cf-b758-f2f1ea467818	Frank St. Leger
97	/artist/22b95e86-0749-4df1-ae29-cd5acfe5a285	Jim Murray

(b) Preview of the result filtered by “male” gender

Table 4.7: Results from 1^{st} reformulation process

PUC-Rio - Certificação Digital Nº 1912709/CA

Since this result was still higher than our β threshold, the process was reapplied. Again, the process grouped the results by predicate and counted the distinct object values. Table 4.8 presents this result.

#	**Predicate**	**Appears**	**Distinct Values**
1	http://dbpedia.org/ontology/birthDate	286	110
2	http://xmlns.com/foaf/0.1/givenName	192	98
3	http://xmlns.com/foaf/0.1/name	184	108
4	http://www.w3.org/2000/01/rdf-schema#label	159	97
5	http://dbpedia.org/ontology/wikiPageID	159	94
6	http://dbpedia.org/ontology/wikiPageRevisionID	159	94
7	http://purl.org/dc/terms/description	153	68
8	http://xmlns.com/foaf/0.1/surname	113	67
9	http://dbpedia.org/ontology/activeYearsStartYear	97	29
10	http://dbpedia.org/ontology/background	92	3
11	http://dbpedia.org/ontology/deathDate	73	46

Table 4.8: Available predicates in the 2^{nd} reformulation process

Continuing with the proposed Π heuristic, the process chose the predicate `http://dbpedia.org/ontology/background`, and a list with 3 available options to choose from was generated, as shown in Table 4.9(a). After the process chose option `2. non_vocal_instrumentalist`, the final result decreased to 23 artists, as shown in Table 4.9(b).

#	**Options**	**Counts**
1	non_performing_personnel	5
2	non_vocal_instrumentalist	23
3	solo_singer	20

(a) Available facets for predicate artist background

#	**Artist partial URI**	**Artist name**
1	/artist/b09ae88f-4156-4caa-b129-1cacb5e1632e	Benny Goodman
2	/artist/a5ee1ebe-a645-45d2-8319-d101fe62e581	Biosphere
3	/artist/fa1de503-aba7-41fa-a1ed-371b3e87a717	Madeon
...	...	...
21	/artist/2f4edec3-4110-4469-af6b-093c2c18b4ff	PeeWee Erwin
22	/artist/79ffeffe-93fd-4d90-9cab-1b0faebecbc1	Steve West
23	/artist/f072de17-d65b-4b8b-a507-5a53658f50de	Jonas Ekdahl

(b) Preview of the result filtered by "non_vocal_instrumentalist" background

Table 4.9: Results from 2^{nd} interaction

Once again, this result was still higher than our β threshold, the process was reapplied. Thus, the process grouped the results by predicate and counted the distinct object values. Table 4.10 presents this result.

Continuing with the proposed Π heuristic, the process chose the predicate `http://dbpedia.org/ontology/alias`, and a list with 9 available options to choose from was generated, as shown in Table 4.11(a). After the process chose option `2. Beyond The Wizards Sleeve`, the final result decreased to

PUC-Rio - Certificação Digital Nº 1912709/CA

#	Predicate	Appears	Distinct Values
1	http://dbpedia.org/ontology/birthDate	90	37
2	http://xmlns.com/foaf/0.1/givenName	69	31
3	http://xmlns.com/foaf/0.1/name	49	23
4	http://www.w3.org/2000/01/rdf-schema#label	45	23
5	http://dbpedia.org/ontology/wikiPageID	45	21
6	http://dbpedia.org/ontology/wikiPageRevisionID	45	21
7	http://purl.org/dc/terms/description	43	16
8	http://dbpedia.org/ontology/activeYearsStartYear	33	14
9	http://xmlns.com/foaf/0.1/surname	31	15
10	http://dbpedia.org/ontology/alias	26	9
11	http://dbpedia.org/ontology/deathDate	24	12
12	http://dbpedia.org/property/occupation	24	9
13	http://dbpedia.org/property/caption	23	10

Table 4.10: Available predicates in the 3^{rd} reformulation process

2 artists. The process finally stopped, since it achieved our β threshold of 15 lines. Hence, the final result was presented to the user in a decreased order of *InfoRank* score, as shown in Table 4.11(b).

#	Options	Counts
1	"King of Swing", "The Professor", "Patriarch of the Clarinet", "Swing's Senior Statesman"	1
2	Beyond The Wizards Sleeve	2
3	Bleep, Cosmic Explorer, E-Man	2
4	Jonas Ekdahl	1
5	Kurtis Rush	2
6	Mustapha 3000	2
7	Topper	1
8	Tram, Frankie	1
9	Zach Smith, ABSIV, ABS4	1

(a) Available facets for predicate artist alias

#	Artist partial URI	Artist name
1	/artist/80cb9f52-04b5-4084-a2eb-6098c91cb48a	Erol Alkan
2	/artist/e3ae5763-2298-40d8-90b8-a85da57e8d06	Kurtis Rush

(b) Preview of the result filtered by "Beyond The Wizards Sleeve" background

Table 4.11: Results from 3^{rd} interaction

PUC-Rio - Certificação Digital Nº 1912709/CA

4.2.4
Applying the Ω heuristic over MusicBrainz

Since the original question resulted on a long result set, the process starts the 1^{st} interaction step. The result set related to SPARQL query on Listing 4.1 is reused, but this time to capture the facets that are candidates to be used for aggregation, regardless the predicate.

As in the previous heuristics, the three-column result set is used and the metrics: *distinct values* and *presence* - related to *predicates* - are calculated. After applied previously defined thresholds α and δ, a list of available facets sorted by decreased distinct values is presented, as shown in Table 4.12.

#	Predicate	Facet	Distinct Values
1	http://xmlns.com/foaf/0.1/gender	male	97
2	http://dbpedia.org/ontology/background	solo_singer	27
3	http://xmlns.com/foaf/0.1/gender	female	26
4	http://dbpedia.org/ontology/background	non_vocal_instrumentalist	25
5	http://dbpedia.org/ontology/background	non_performing_personnel	5

Table 4.12: Available facets in the 1^{st} reformulation process

Instead of selecting first a predicate and then a facet, all available facets are analysed together. Recall the Ω heuristic, which the approach is to select the most **embracing** facet, regardless the predicate. Thus, the most embracing facet is `male`, as shown in Table 4.12. After the process chose option `1. male`, the final result decreased to 97 artists, as shown in Table 4.13.

#	Artist partial URI	Artist name
1	/artist/b09ae88f-4156-4caa-b129-1cacb5e1632e	Benny Goodman
2	/artist/27b0750a-7318-4075-9470-43b82d454ea0	Gustav Leonhardt
3	/artist/2c69465c-0f76-45ce-90a2-1ed0fdacc997	CeeLo Green
...	...	...
95	/artist/5d0b474d-d24a-4b5c-8a67-7a79f7f9949a	Judd Woldin
96	/artist/e00871b0-f6b5-41cf-b758-f2f1ea467818	Frank St. Leger
97	/artist/22b95e86-0749-4df1-ae29-cd5acfe5a285	Jim Murray

Table 4.13: Preview of the result filtered by "male"

Since this result was still higher than our β threshold, the process was reapplied. Again, the process analysed together all available facets generating a list of the selectable facets sorted by decreased distinct values, as shown in Table 4.14.

#	Predicate	Facet	Distinct Values
1	http://dbpedia.org/ontology/background	non_vocal_instrumentalist	23
2	http://dbpedia.org/ontology/background	solo_singer	20
3	http://dbpedia.org/ontology/background	non_performing_personnel	5

Table 4.14: Available facets in the 2^{nd} reformulation process

PUC-Rio - Certificação Digital Nº 1912709/CA

Continuing with the proposed Ω heuristic, the process chose option `1. non_vocal_instrumentalist`. The final result decreased to 23 artists, as shown in Table 4.15.

#	Artist partial URI	Artist name
1	/artist/b09ae88f-4156-4caa-b129-1cacb5e1632e	BennyGoodman
2	/artist/a5ee1ebe-a645-45d2-8319-d101fe62e581	Biosphere
3	/artist/fa1de503-aba7-41fa-a1ed-371b3e87a717	Madeon
...	...	...
21	/artist/2f4edec3-4110-4469-af6b-093c2c18b4ff	PeeWeeErwin
22	/artist/79ffeffe-93fd-4d90-9cab-1b0faebecbc1	SteveWest
23	/artist/f072de17-d65b-4b8b-a507-5a53658f50de	JonasEkdahl

Table 4.15: Preview of the result filtered by "non_vocal_instrumentalist"

Once again, this result was still higher than our β threshold, the process was reapplied. Again, the process analysed together all available facets generating a list of the selectable facets sorted by decreased distinct values, as shown in Table 4.16.

#	Predicate	Facet	Distinct Values
1	http://dbpedia.org/property/occupation	Musician	4
2	http://dbpedia.org/property/caption	Erol Alkan Live	2
3	http://dbpedia.org/property/caption	Geir Jenssen performing at Creative Camp...	2
...	...	...	...
12	http://dbpedia.org/property/caption	Madeon in 2015	1
13	http://dbpedia.org/property/caption	Pipien on stage with The Black Crowes at...	1
14	http://dbpedia.org/property/caption	Randy Napoleon	1
...	...	...	...
21	http://dbpedia.org/property/occupation	DJ	1
22	http://dbpedia.org/property/occupation	Drummer	1
23	http://dbpedia.org/property/occupation	Drummer, percussionist, songwriter	1
...	...	...	...
26	http://dbpedia.org/property/occupation	Songwriter	1
27	http://dbpedia.org/property/occupation	musician	1
28	http://dbpedia.org/property/occupation	record producer	1

Table 4.16: Available facets in the 3^{rd} reformulation process

Continuing with the proposed Ω heuristic, the process chose the option `1. Musician`. The final result decreased to 4 artists, as shown in Table 4.17. The process finally stopped, since it achieved our β threshold of 15 lines. Hence, the final result was presented to the user in a decreased order of *InfoRank* score, as shown in Table 4.17.

#	Artist partial URI	Artist name
1	/artist/aec79f86-6547-4172-8ff7-a31e701135ac	Troy Donockley
2	/artist/b0cb4c80-308f-4bbd-8924-a0a632a83e0a	Randy Napoleon
3	/artist/696e9995-acbf-40b4-9346-f79d0c3cc2f1	Stuart Smith
4	/artist/f072de17-d65b-4b8b-a507-5a53658f50de	Jonas Ekdahl

Table 4.17: Preview of the result filtered by "Musician"

PUC-Rio - Certificação Digital Nº 1912709/CA

4.3 IMDb Results

4.3.1 Sample Query

In this section, we use a query adapted from the Coffman's benchmark over IMDb: *"Which movies did Denzel Washington starred?"*. The initial process generated the SPARQL query bellow, ranking the results by the *InfoRank* score.

```
prefix imdb: <http://www.imdb.com/>
prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#>
prefix quira: <http://www.quira.org/>

select distinct ?movie ?label ?inforank
where {
  ?movie a imdb:Movie .
  ?movie imdb:actor ?actor .
  ?movie rdfs:label ?label .
  ?movie quira:inforank ?inforank .
  ?actor imdb:name ``Denzel Washington'' .
}
order by desc(?inforank)
```

Table 4.18 shows a preview of the original result. Note that this SPARQL query returned 49 movies starred by Denzel Washington.

#	Movie URI	Movie title
1	http://www.imdb.com/work/1996688	Malcolm X
2	http://www.imdb.com/work/1592464	American Gangster
3	http://www.imdb.com/work/2354723	Unstoppable
...	...	...
47	http://www.imdb.com/work/1675254	Champs
48	http://www.imdb.com/work/2255730	The Equalizer
49	http://www.imdb.com/work/2356601	Uptown Saturday Night

Table 4.18: Preview of the original SPARQL result

Again, we used the maximum of 15 lines as the β threshold to indicate when the process should be applied. In this example, it would be a list with a maximum of 15 movies. The stop condition is not valid and the process must go on. In the following subsections, we will be able to compare the three proposed heuristics.

PUC-Rio - Certificação Digital Nº 1912709/CA

4.3.2
Applying the Σ heuristic over IMDb

As the stop condition was invalid in the previous step, the process reformulated the SPARQL query to capture the predicates used for aggregation.

```
prefix imdb: <http://www.imdb.com/>

select distinct ?movie ?predicate ?object
where {
  {
    select ?movie
    where {
      ?movie a imdb:Movie .
      ?movie imdb:actor ?actor .
      ?actor imdb:name ''Denzel Washington''
    }
  }
  .
  {
    select ?movie ?predicate ?object
    where {
      ?movie ?predicate ?object .
      filter(isLiteral(?object))
    }
  }
}
```

Listing 4.2: Transforming single-column into three-column result set

Once again, the process grouped the results by predicate and counted the distinct object values. Table 4.19 presents part of the available predicates. Following the proposed Σ heuristic, the process chose the predicate `http://www.imdb.com/label`, which refers to the production company of the film.

#	Predicate	Distinct Values
1	http://www.imdb.com/tag	2408
2	http://www.imdb.com/release_dates	1211
3	http://www.imdb.com/quotes	936
...	...	...
36	http://www.imdb.com/novel	14
37	http://www.imdb.com/label	10
38	http://www.imdb.com/number_of_chapter_stops	8
...	...	...
66	http://www.imdb.com/interviews	1
67	http://www.imdb.com/master_format	1
68	http://www.imdb.com/quality_program	1

Table 4.19: Available predicates

PUC-Rio - Certificação Digital Nº 1912709/CA

Hence, the process now chooses the most embracing facet from the options presented in Table 4.20(a), which corresponds to the 10 distinct values of the predicate `http://www.imdb.com/label`. Following the Σ heuristic, the process chose option `1. Columbia/Tristar` film label, which has one of the highest movie counts.

The final result had only 5 movies, and the process stopped at this point, since it achieved a reasonably compact result to present to the user ($5 < \beta = 15$). The final result was presented to the user in a decreased order of *InfoRank* score, as shown in Table 4.20(b).

#	**Options**	**Counts**
1	Columbia/Tristar	5
2	Encore	5
3	Warner Home Video	3
4	MCA/Universal Home Video	2
5	Paramount	2
6	20th Century Fox Home Entertainment	1
7	Hollywood Pictures	1
8	Philips	1
9	Pioneer	1
10	RCA/Columbia	1

(a) Available facets for predicate movie label

#	**Movie URI**	**Movie title**
1	http://www.imdb.com/work/2095826	Philadelphia
2	http://www.imdb.com/work/1824613	Glory
3	http://www.imdb.com/work/2031102	Much Ado About Nothing
4	http://www.imdb.com/work/2020598	Mississippi Masala
5	http://www.imdb.com/work/1731607	Devil in a Blue Dress

(b) The final result presented to the user

Table 4.20: Results from 1^{st} interaction

PUC-Rio - Certificação Digital Nº 1912709/CA

4.3.3 Applying the Π heuristic over IMDb

Since the original question resulted on a long result set, the process starts the 1^{st} interaction step. The result set related to SPARQL query on Listing 4.2 is reused to capture the predicates that are candidates to be used for aggregation.

Basically, the three-column result set is grouped by *predicate*, and the metrics: *distinct values* and *presence* are calculated. These two metrics are related to previously defined thresholds α and δ, respectively. Table 4.21 presents the results.

Recall the Π heuristic, which the approach is to select the most **embracing** facet from the most **restrictive** predicate. In the Table 4.21, the most restrictive predicate is `http://www.imdb.com/color_info`. And the most em-

#	Predicate	Appears	Distinct Values
1	http://www.imdb.com/sound_mix	85	8
2	http://www.imdb.com/year	47	27
3	http://www.imdb.com/color_info	47	2
...	...	...	...
11	http://www.imdb.com/budget	35	28
12	http://www.imdb.com/release_date	25	24
13	http://www.imdb.com/sound_encoding	25	4
...	...	...	...
22	http://www.imdb.com/video_standard	23	2
23	http://www.imdb.com/label	22	10
24	http://www.imdb.com/length	21	19

Table 4.21: Available predicates in the 1^{st} reformulation process

bracing facet is `Color`, as shown in Table 4.22(a). After the process chose option `2. Color`, the final result decreased to 46 movies, as shown in Table 4.22(b).

#	Options	Counts
1	Black and White	1
2	Color	46

(a) Available facets for predicate movie color_info

#	Movie partial URI	Movie name
1	/work/1996688	Malcolm X
2	/work/1592464	American Gangster
3	/work/2354723	Unstoppable
...	...	...
44	/work/2232130	The 100 Best Black Movies (Ever)
45	/work/1675254	Champs
46	/work/2255730	The Equalizer

(b) Preview of the result filtered by “Color” color_info

Table 4.22: Results from 1^{st} reformulation process

Since this result was still higher than our β threshold, the process was reapplied. Again, the process grouped the results by predicate and counted the distinct object values. Table 4.23 presents this result.

Continuing with the proposed Π heuristic, the process chose the predicate `http://www.imdb.com/video_standard` and the most embracing facet from the options presented in Table 4.24(a).

After selecting the option `1. NTSC`, the final result decreased to 14 artists, as shown in Table 4.24(b). The process finally stopped, since it achieved our β threshold of 15 lines. Hence, the final result was presented to the user in a decreased order of *InfoRank* score, as shown in Table 4.24(b).

PUC-Rio - Certificação Digital Nº 1912709/CA

#	Predicate	Appears	Distinct Values
1	http://www.imdb.com/admissions	85	85
2	http://www.imdb.com/sound_mix	81	8
3	http://www.imdb.com/plot	70	70
4	http://www.imdb.com/opening_weekend	69	69
...	...	...	...
11	http://www.imdb.com/year	44	27
12	http://www.imdb.com/release_date	24	23
13	http://www.imdb.com/sound_encoding	24	4
14	http://www.imdb.com/video_standard	22	2
...	...	...	...
23	http://www.imdb.com/label	21	10
24	http://www.imdb.com/length	20	19
25	http://www.imdb.com/certification	20	6
26	http://www.imdb.com/alternate_versions	19	19

Table 4.23: Available predicates in the 2^{nd} reformulation process

#	Options	Counts
1	NTSC	14
2	PAL	8

(a) Available facets for predicate movie video_standard

#	Movie partial URI	Movie name
1	/work/1996688	Malcolm X
2	/work/2095826	Philadelphia
3	/work/1824613	Glory
4	/work/1701013	Crimson Tide
...	...	...
11	/work/1844055	He Got Game
12	/work/1702441	Cry Freedom
13	/work/2031102	Much Ado About Nothing
14	/work/1731607	Devilina Blue Dress

(b) Preview of the result filtered by "NTSC" video_standard

Table 4.24: Results from 2^{nd} interaction

PUC-Rio - Certificação Digital Nº 1912709/CA

4.3.4 Applying the Ω heuristic over IMDb

Since the original question resulted on a long result set, the process starts the 1^{st} interaction step. The result set related to SPARQL query on Listing 4.2 is reused, but this time to capture the facets that are candidates to be used for aggregation, regardless the predicate.

As in the previous heuristics, the three-column result set is used and the metrics: *distinct values* and *presence* - related to *predicates* - are calculated. After applied previously defined thresholds α and δ, a list of available facets sorted by decreased distinct values is presented, as shown in Table 4.25.

Instead of selecting first an predicate and then a facet, all available facets are analysed together. Recall the Ω heuristic, which the approach is to select

#	Predicate	Facet	Distinct Values
1	http://www.imdb.com/color_info	Color	46
2	http://www.imdb.com/sound_mix	Dolby Digital	29
3	http://www.imdb.com/sound_mix	SDDS	20
...	...	...	...
12	http://www.imdb.com/label	Columbia/Tristar	5
13	http://www.imdb.com/label	Encore	5
16	http://www.imdb.com/label	Warner Home Video	3
...	...	...	...
24	http://www.imdb.com/label	20th Century Fox Home Entertainment	1
25	http://www.imdb.com/label	Hollywood Pictures	1
26	http://www.imdb.com/label	Philips	1
...	...	...	...
30	http://www.imdb.com/certification	PG-13	1
31	http://www.imdb.com/sound_encoding	Analog	1
32	http://www.imdb.com/color_info	Black and White	1

Table 4.25: Available facets in the 1^{st} reformulation process

the most **embracing** facet, regardless the predicate. Thus, the most embracing facet is `Color`, as shown in Table 4.25. After the process chose option `1. Color`, the final result decreased to 46 movies, as shown in Table 4.26.

#	Movie partial URI	Movie name
1	/work/1996688	Malcolm X
2	/work/1592464	American Gangster
3	/work/2354723	Unstoppable
...	...	...
44	/work/2232130	The 100 Best Black Movies (Ever)
45	/work/1675254	Champs
46	/work/2255730	The Equalizer

Table 4.26: Preview of the result filtered by "Color"

Since this result was still higher than our β threshold, the process was reapplied. Again, the process analysed together all available facets generating a list of the selectable facets sorted by decreased distinct values, as shown in Table 4.27.

Continuing with the proposed Ω heuristic, the process chose option `1. Dolby Digital`. The final result decreased to 28 artists, as shown in Table 4.28.

Once again, this result was still higher than our β threshold, the process was reapplied. Again, the process analysed together all available facets generating a list of the selectable facets sorted by decreased distinct values, as shown in Table 4.29.

Continuing with the proposed Ω heuristic, the process chose the option `1. NTSC`. The final result decreased to 9 artists, as shown in Table 4.30. The process finally stopped, since it achieved our β threshold of 15 lines. Hence, the final result was presented to the user in a decreased order of *InfoRank* score, as shown in Table 4.30.

PUC-Rio - Certificação Digital Nº 1912709/CA

#	Predicate	Facet	Distinct Values
1	http://www.imdb.com/sound_mix	Dolby Digital	28
2	http://www.imdb.com/sound_mix	SDDS	19
3	http://www.imdb.com/sound_mix	DTS	16
...	...	...	...
10	http://www.imdb.com/certification	R	9
12	http://www.imdb.com/sound_encoding	Digital	9
12	http://www.imdb.com/video_standard	PAL	8
...	...	...	...
18	http://www.imdb.com/label	Encore	5
18	http://www.imdb.com/label	Columbia/Tristar	4
20	http://www.imdb.com/label	Warner Home Video	3
...	...	...	...
28	http://www.imdb.com/label	Hollywood Pictures	1
29	http://www.imdb.com/label	Philips	1
30	http://www.imdb.com/label	Pioneer	1

Table 4.27: Available facets in the 2^{nd} reformulation process

#	Movie partial URI	Movie name
1	/work/1996688	Malcolm X
2	/work/1592464	American Gangster
3	/work/2354723	Unstoppable
...	...	...
26	/work/2292136	The Preacher's Wife
27	/work/2079243	Out of Time
28	/work/1731607	Devilina Blue Dress

Table 4.28: Preview of the result filtered by “Dolby Digital”

PUC-Rio - Certificação Digital Nº 1912709/CA

4.4 Compression Rate

In this section, we propose a discussion over the compiled results of each heuristic over both datasets. Also, we decided to use a metric *Compression Rate* to help us compare the obtained results. The metric is defined as follows:

1. γ = compression rate
2. η = # lines of the initial result set
3. κ = # lines of the final result set
4. $\gamma = 1 - \kappa / \eta$

As we can see, when $\kappa \sim \eta$ the compression is very low. On the other hand, when $\kappa <<< \eta$ the compression is very high. It is important to recall that variable β guarantees the final result set will have a maximum number of lines. Thus, this behavior will be reflected on the analysis in cases where the selected facet does not reduce enough the result set.

#	Predicate	Facet	Distinct Values
1	http://www.imdb.com/video_standard	NTSC	9
2	http://www.imdb.com/certification	R	7
3	http://www.imdb.com/sound_encoding	Digital/AC-3/Analog	6
...	...	...	...
13	http://www.imdb.com/official_retail_price	$ 29.98	1
14	http://www.imdb.com/official_retail_price	$ 34.95	1
15	http://www.imdb.com/official_retail_price	$ 39.95	1
...	...	...	...
23	http://www.imdb.com/label	20th Century Fox ...	1
24	http://www.imdb.com/label	Columbia/Tristar	1
25	http://www.imdb.com/label	Paramount	1

Table 4.29: Available facets in the 3^{rd} reformulation process

#	Movie partial URI	Movie name
1	/work/1996688	Malcolm X
2	/work/1701013	Crimson Tide
3	/work/1786796	Fallen
4	/work/1698514	Courage Under Fire
5	/work/2241704	The Bone Collector
6	/work/2289751	The Pelican Brief
7	/work/2366980	Virtuosity
8	/work/1844055	He Got Game
9	/work/1731607	Devilina Blue Dress

Table 4.30: Preview of the result filtered by "NTSC"

PUC-Rio - Certificação Digital Nº 1912709/CA

Positioned at the end of this section, Tables 4.33 and 4.34 have the same header. The column `IRS` means *Initial Result Set* and is related to the number of lines from the original result set. The column `FRS` means *Final Result Set* and is related to the number of lines from the compressed result set. The column `Facets` has the selected predicate and facet separated by pipe `|`. It is important to mention that the number between parenthesis is the number unique subjects. In other words, the number of lines of the result set filtered by facet. Also, this column may have multiple facets displayed in multi-lines. The number of lines must reflect the number of `Steps` the process took.

The results over MusicBrainz are presented on Table 4.33. Seven questions are listed on it. Ten questions were originally tested over dataset. But three of them had no selected predicate. This happened because the process could not find a single predicate that respects the defined thresholds α and δ. For instance, the question *"What are the songs performed by Aretha Franklin?"* has the following predicates:

– http://purl.org/ontology/mo/track_number (54)

– http://www.w3.org/2000/01/rdf-schema#label (1120)

– http://purl.org/ontology/mo/duration (1881)

In these cases, all the three numbers in parenthesis are higher than α. The original result set had 2945 lines and the final result set had β lines. Even when the process does not perform any step, it ranks the results using *InfoRank* metric and return the results filtered by β to the user. But we decided to not consider these cases anyway.

Suppose we decided to relax the thresholds, the three selectable predicates would still not be interesting, because of the lack of meaning. One approach that might be tested in the future is the predicate taxonomy. The user would be able to inform that specific predicates have multi-level analysis. In a second version, the system would be able to infer possible multi-level predicates. So, the process would try to group information based on the most higher level.

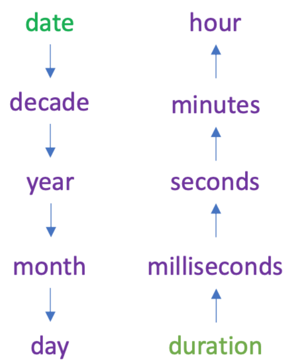


Figure 4.3: Predicate taxonomy examples

As shown in Figure 4.3, the predicate `mo:duration` would have an associate taxonomy. Thus, in our song context, it might be interesting to group songs by minutes. Based on Music Ontology Specification[4], the predicate `mo:duration` represents the duration of a track or a signal in ms. Analysing the songs by Aretha Franklin, her longest song is "Amazing Grace" with 10min and 48s ($\sim$11min).

In Table 4.31, the results for predicate `mo:duration` analysis in minutes are presented. We can affirm that all proposed heuristics would apply one of the ten possible length-facets.

Continuing the analysis, the third question *"Which artists played on the same groups that David Bowie was member of?"* had the original result set length 17 which is very close to $\beta(= 15)$. Thus, the obtained compression rate applying heuristics Π and Ω were the worst. But the predicate `foaf:gender` seems much more interesting than `dbo:wikiPageID`, selected when applying Σ heuristic. Hence, the Compression Rate metric by itself does not mean all. We still need to analyse the path over K-D tree performed by the process.

[4] http://musicontology.com/specification/#term-duration

PUC-Rio - Certificação Digital Nº 1912709/CA

#	**Duration**	**Total Songs**
1	11.0	9
2	10.0	6
3	9.0	21
4	8.0	28
5	7.0	69
6	6.0	157
7	5.0	337
8	4.0	781
9	3.0	1110
10	2.0	399

Table 4.31: Results for `mo:duration` analysis in minutes

Considering a non-technical user, we can discuss meaningfulness of predicates chosen and facets selected. The first question *"Which artists were born on May 30th?"* navigated through meaningful predicates and facets, also presented excellent compression rate. Only the Π heuristic chose a questionable predicate (`dbo:alias`) in its 3^{rd} step. We affirm that because this predicate should be almost unique by artist. Thus, the process is selecting an specific artist instead of selecting a property that few artists have in common. To solve this problem, the users will be able to exclude undesired predicates in future work.

A curious result was found on seventh question *"Which bands broke up in 2010?"*. In all three heuristics, the process selected meaningful predicates but useless. This happened because the selected predicates were related to the original question. Hence, selecting `group_or_band` and `2010`, compressed almost nothing when comparing the original and final result sets. The final compression rate was good again because of the β threshold. Without this threshold, the compression would be near zero.

The results over IMDb are presented on Table 4.34. Another seven questions are listed on it. In this case, the most meaningful predicates chosen and facets selected were in the first question *"Which movies did Denzel Washington starred?"*. Although the compression rates were good, when compared to the other questions, the compression rates were the worst (together with the sixth question). This happened because all other questions had no restrict facet and the original result set was very long. Thus, the $\kappa <<< \eta$ since $\kappa = \beta = 15$.

We decided to investigate the lack of predicates with restrictive facets. The questions used on dataset were related to `imdb:Movie`, `imdb:Actor` and `imdb:Actress`. We joined resources `imdb:Actor` and `imdb:Actor` in this investigation and called then *Artists*.

PUC-Rio - Certificação Digital Nº 1912709/CA

```
prefix imdb: <http://www.imdb.com/>

select ?pred (count(distinct ?object) as ?dist_values)
where {
  ?movie a imdb:Movie .
  ?movie ?pred ?object .
  filter(isLiteral(?object)) .
}
group by ?pred
order by desc(?dist_values)
```

Listing 4.3: Investigation over `imdb:Movie`

In Listing 4.3, the SPARQL query selects the unique predicates related to instances of `imdb:Movie` and count their distinct literal values. Recall the α threshold, and compare its value with the results shown in Table 4.32(a). From 68 predicates, only 15 are available (options 54 to 68). Also, recall the δ threshold, which is responsible to filter predicates based on its appearance rate over the result set. Thus, the set of selectable predicates is small and their meaningfulness is also low. Except `imdb:category`, all predicates are related to technical information about the movies.

In Listing 4.4, the SPARQL query selects the unique predicates related to instances of *Artists* and count their distinct literal values. Once again, very similar to previous case, there are few available predicates. In Table 4.32(b), from 26 predicates, only one is available (option 26).

```
prefix imdb: <http://www.imdb.com/>

select ?pred (count(distinct ?object) as ?dist_values)
where {
  ?artist a ?klass .
  filter(?klass in (imdb:Actor, imdb:Actress)) .
  ?artist ?pred ?object .
  filter(isLiteral(?object)) .
}
group by ?pred
order by desc(?dist_values)
```

Listing 4.4: Investigation over *Artists*

PUC-Rio - Certificação Digital Nº 1912709/CA

#	**Predicate**	**Distinct Values**
1	rdfs:label	512491
2	imdb:title	512469
3	imdb:release_dates	274127
...	...	...
53	imdb:subtitles	12
54	imdb:analog_left	8
55	imdb:picture_format	8
56	imdb:sound_encoding	8
57	imdb:digital_sound	7
58	imdb:analog_right	6
59	imdb:category	4
60	imdb:color_information	4
61	imdb:status_of_availablility	4
62	imdb:video_standard	4
63	imdb:close_captions-teletext-ld-g	3
64	imdb:disc_format	3
65	imdb:disc_size	3
66	imdb:master_format	3
67	imdb:color_info	2
68	imdb:quality_program	1

(a) Predicate investigation over `imdb:Movie`

#	**Predicates**	**Distinct Values**
1	imdb:name	2192378
2	rdfs:label	2191943
3	imdb:trivia	495432
4	imdb:aka	470130
5	imdb:other_works	305969
...	...	...
22	imdb:salary_history	5384
23	imdb:biographical_movies	4648
24	imdb:portrayed_in	3585
25	imdb:height	444
26	imdb:gender	2

(b) Predicate investigation over *Artists*

Table 4.32: Investigation results over IMDb predicates

PUC-Rio - Certificação Digital Nº 1912709/CA

		MusicBrainz			
Question	**1**	**Which artists were born on May 30th?**			
Heuristic	Steps	Facets	IRS	FRS	Compression Rate
Σ	2	dbo:background | solo_singer (27); dbp:occupation | Singer-songwriter (5);	123	5	95,93%
Π	3	foaf:gender | male (97); dbo:background | non_vocal_instrumentalist (23); dbo:alias | Beyond The Wizards Sleeve (2);	123	2	98,37%
Ω	3	foaf:gender | male (97); dbo:background | non_vocal_instrumentalist (23); dbp:occupation | Musician (4);	123	4	96,75%
Question	**2**	**Which songs by Miles Davis are longer than 20 minutes?**			
Heuristic	Steps	Facets	IRS	FRS	Compression Rate
Σ	1	mo:track_number | 1 (48)	89	15	83,15%
Π	1	mo:track_number | 1 (48)	89	15	83,15%
Ω	1	mo:track_number | 1 (48)	89	15	83,15%
Question	**3**	**Which artists played on the same groups that David Bowie was member of?**			
Heuristic	Steps	Facets	IRS	FRS	Compression Rate
Σ	1	dbo:wikiPageID | 1515176 (1);	17	1	94,12%
Π	1	foaf:gender | male (14);	17	14	17,65%
Ω	1	foaf:gender | male (14);	17	14	17,65%
Question	**4**	**What are the albums from Michael Jackson?**			
Heuristic	Steps	Facets	IRS	FRS	Compression Rate
Σ	1	dbp:years | –05-24 (1)	23	1	95,65%
Π	2	dbp:writingCredits | yes (20); dbp:artist | Michael Jackson (10);	23	10	56,52%
Ω	2	dbp:writingCredits | yes (20); dbp:award | Gold (10);	23	10	56,52%
Question	**5**	**What are the albums from Kraftwerk?**			
Heuristic	Steps	Facets	IRS	FRS	Compression Rate
Σ	1	dbp:writer | Hütter (2)	13	2	84,62%
Π	1	dbp:headline | Side one (6)	13	6	53,85%
Ω	1	dbp:type | studio (7)	13	7	46,15%
Question	**6**	**Which artists were born on September, 1964?**			
Heuristic	Steps	Facets	IRS	FRS	Compression Rate
Σ	2	dbo:background | solo_singer (18); foaf:surname | Anastasio (1);	66	1	98,48%
Π	2	foaf:gender | male (48); dbo:background | non_vocal_instrumentalist (14);	66	14	78,79%
Ω	2	foaf:gender | male (48); dbo:background | non_vocal_instrumentalist (14);	66	14	78,79%
Question	**7**	**Which bands broke up in 2010?**			
Heuristic	Steps	Facets	IRS	FRS	Compression Rate
Σ	2	dbo:background | group_or_band (236); dbo:activeYearsEndYear | 2010 (236);	238	15	93,70%
Π	2	dbo:background | group_or_band (236); dbo:activeYearsEndYear | 2010 (236);	238	15	93,70%
Ω	2	dbo:activeYearsEndYear | 2010 (238); dbo:background | group_or_band (236);	238	15	93,70%

Table 4.33: Compression Rate comparison over Music Brainz

PUC-Rio - Certificação Digital Nº 1912709/CA

PUC-Rio - Certificação Digital Nº 1912709/CA

		IMDb			
Question	**1**	**Which movies did Denzel Washington starred?**			
Heuristic	Steps	Facets	IRS	FRS	Compression Rate
Σ	1	imdb:label | Columbia/Tristar (5);	49	5	89,80%
Π	2	imdb:color_info | Color (46); imdb:video_standard | NTSC (14);	49	14	71,43%
Ω	3	imdb:color_info | Color (46); imdb:sound_mix | Dolby Digital (28); imdb:video_standard | NTSC (9);	49	9	81,63%
Question	**2**	**Which movies are available in spanish language?**			
Heuristic	Steps	Facets	IRS	FRS	Compression Rate
Σ	1	imdb:color_info | Color (67933);	77670	15	99,98%
Π	1	imdb:color_info | Color (67933);	77670	15	99,98%
Ω	1	imdb:color_info | Color (67933);	77670	15	99,98%
Question	**3**	**Which actors or actresses were born on May 30th?**			
Heuristic	Steps	Facets	IRS	FRS	Compression Rate
Σ	1	imdb:gender | Male (409);	595	15	97,48%
Π	1	imdb:gender | Male (409);	595	15	97,48%
Ω	1	imdb:gender | Male (409);	595	15	97,48%
Question	**4**	**Which movies were released in 2000?**			
Heuristic	Steps	Facets	IRS	FRS	Compression Rate
Σ	2	imdb:color_info | Color (5815); imdb:year | 2000 (5815);	7206	15	99,79%
Π	2	imdb:year | 2000 (7488); imdb:color_info | Color (5815);	7206	15	99,79%
Ω	2	imdb:year | 2000 (7488); imdb:color_info | Color (5815);	7206	15	99,79%
Question	**5**	**Which movies were produced in Brazil?**			
Heuristic	Steps	Facets	IRS	FRS	Compression Rate
Σ	1	imdb:color_info | Color (4432);	7016	15	99,79%
Π	1	imdb:color_info | Color (4432);	7016	15	99,79%
Ω	1	imdb:color_info | Color (4432);	7016	15	99,79%
Question	**6**	**Which movies were produced in Brazil in 2010?**			
Heuristic	Steps	Facets	IRS	FRS	Compression Rate
Σ	2	imdb:color_info | Color (74); imdb:year | 2000 (74);	91	15	83,52%
Π	2	imdb:year | 2000 (96); imdb:color_info | Color (74);	91	15	83,52%
Ω	2	imdb:year | 2000 (96); imdb:color_info | Color (74);	91	15	83,52%
Question	**7**	**Which brazilian artists starred foreign movies?**			
Heuristic	Steps	Facets	IRS	FRS	Compression Rate
Σ	1	imdb:gender | Male (733);	1261	15	98,81%
Π	1	imdb:gender | Male (733);	1261	15	98,81%
Ω	1	imdb:gender | Male (733);	1261	15	98,81%

Table 4.34: Compression Rate comparison over IMDb

5
Conclusion and Future Work

The main contribution of this study was the definition of a process - called *Query Answer Modification Process* - based on simple heuristics and parameterized thresholds that reorganizes the query answer improving the quality of the user's experience. This study addressed open-ended questions since specific questions do not generate long result sets. Thus, the summarization task would not have the expected effect.

To validate the proposed process, sample queries from QALD challenge and Coffman's benchmark were used. These queries were applied over RDF versions of MusicBrainz and IMDb, respectively. Also, a *Compression Rate* metric was defined enabling comparison and discussion over compiled results.

The initial research involving the *Query Answer Modification Process* was published in the Proceedings of the XXXV Brazilian Symposium on Databases – SBBD. An extended version of this paper was submitted to the Journal of Information and Data Management (JIDM) and is under review at the time of this writing.

A straightforward suggestion of future work would be allowing the users to provide a list of ignored predicates, or allowing them to dynamically exclude undesired predicates from the available list in each step of the process. Thus, lack of meaning predicates would be ignored and the results tend to improve. Also in this context, registering users' feedback after each processed query answer modification would provide automatic fill of this list.

A follow-up study to this dissertation would develop and apply a questionnaire over users interested in using the system. As a result, we would be able to segment users by preferences creating profiles. These profiles would enrich the qualitative discussion on Chapter 4. Likewise, these profiles could be used to apply specific heuristic based on the user.

Another interesting suggestion would be allowing the users to inform that specific predicates can have multi-level analysis. So, the system would be able to aggregate over an embracing level predicate. In a second version, together with the initial exploration of RDF Knowledge Base to calculate frequencies, the system would register data types of the available predicates and automatically infer the related taxonomy.

PUC-Rio - Certificação Digital Nº 1912709/CA

Finally, the development of a user interface similar to GraFa [9] would be interesting since it proposes navigation through predicates and facets like our process does. As well, users of all types (beginners or experienced) would be able to use and evaluate our process.

PUC-Rio - Certificação Digital Nº 1912709/CA

Bibliography

[1] Berners-Lee, T., Hendler, J., and Lassila, O. The semantic web; a new form of web content that is meaningful to computers will unleash a revolution of new possibilities. *Scientific American* (May 2001).

[2] Coffman, J., and Weaver, A. C. A framework for evaluating database keyword search strategies. In *Proceedings of the 19th ACM international conference on Information and knowledge management* (2010), pp. 729–738.

[3] Dalianis, H., and Hovy, E. Aggregation in natural language generation. In *Trends in Natural Language Generation An Artificial Intelligence Perspective*, J. G. Carbonell, J. Siekmann, G. Goos, J. Hartmanis, J. Leeuwen, G. Adorni, and M. Zock, Eds., vol. 1036. Springer Berlin Heidelberg, Berlin, Heidelberg, 1996, pp. 88–105.

[4] Deutch, D., Frost, N., and Gilad, A. Provenance for natural language queries. *Proceedings of the VLDB Endowment 10*, 5 (Jan. 2017), 577–588.

[5] Diefenbach, D., Lopez, V., Singh, K., and Maret, P. Core techniques of question answering systems over knowledge bases: a survey. *Knowledge and Information Systems 55*, 3 (June 2018), 529–569.

[6] Diefenbach, D., Tanon, T. P., Singh, K., and Maret, P. Question Answering Benchmarks for Wikidata. In *ISWC 2017* (Vienne, Austria, Oct. 2017).

[7] Franz, T., Schultz, A., Sizov, S., and Staab, S. TripleRank: Ranking semantic web data by tensor decomposition. In *Lecture Notes in Computer Science*. Springer Berlin Heidelberg, 2009, pp. 213–228.

[8] Menendez, E. S., Casanova, M. A., Leme, L. A. P., and Boughanem, M. Novel node importance measures to improve keyword search over rdf graphs. In *International Conference on Database and Expert Systems Applications* (2019), Springer, pp. 143–158.

[9] Moreno-Vega, J., and Hogan, A. Grafa: Faceted search & browsing for the wikidata knowledge graph. In *International Semantic Web Conference* (2018).

PUC-Rio - Certificação Digital Nº 1912709/CA

[10] Petzka, H., Stadler, C., Katsimpras, G., Haarmann, B., and Lehmann, J. Benchmarking faceted browsing capabilities of triplestores. In *Proceedings of the 13th International Conference on Semantic Systems* (New York, NY, USA, 2017), Semantics2017, Association for Computing Machinery, p. 128–135.

[11] Webber, B. L. Questions, answers and responses: Interacting with knowledge-base systems. In *Topics in Information Systems*. Springer New York, 1986, pp. 365–402.

[12] Wei, B., Liu, J., Zheng, Q., Zhang, W., Fu, X., and Feng, B. A survey of faceted search. *Journal of Web Engineering 12* (02 2013), 41–64.

PUC-Rio - Certificação Digital Nº 1912709/CA