4
Processo proposto de Recomendação de Fontes RDF

No processo de publicação de dados, os responsáveis pela publicação devem ter conhecimento de fontes existentes onde podem encontrar recursos de interesse, os quais podem ser usados ou conectados. Porém com o crescimento do número de fontes publicadas na *Linked Data*, identificar recursos e conjunto de dados se torna problemático [8, 15]. Consequentemente, os responsáveis pela publicação de dados apenas conectam os seus dados com fontes de dados populares (como DBPedia e Geonames[1]) [15], mas esta solução nem sempre é a melhor como nos seguintes cenários, por exemplo:

- Quando os dados do domínio são altamente especializados e não abrangidos por essas fontes populares.
- Quando as diferentes partes dos dados publicados são abrangidos por várias fontes de dados. Por exemplo, uma fonte contém referências a publicações científicas sobre ciências da computação (descrito por DBLP[2]) e à recursos relacionados ao domínio da medicina (descrito por PubMed[3]).

Existem iniciativas no nível dos metadados para gerar descritores das fontes de dados. Os descritores são fornecidos pelos responsáveis da publicação usando o vocabulário VoiD. No entanto, esses descritores podem não ser suficientes, pois não consideram a distribuição das instâncias na fonte de dados [24].

Por conseguinte, para escolher uma fonte de dados adequada com recursos apropriados e conectá-los, é necessário analisar dados armazenados no nível de instância. As informações a respeito das instâncias ajudam na descrição e classificação de uma fonte de dados. Porém, muitas vezes obter essa informação diretamente da fonte e analisá-la não é possível, devido ao tamanho dos dados e restrições de acesso da mesma fonte [15].

Desta forma, torna-se necessário o desenvolvimento de um conjunto de mecanismos para recomendação de fontes de dados RDF. Estes mecanismos são

[1] http://wwww.geonames.org
[2] http://dblp.l3s.de
[3] http://wwww.ncbi.nlm.nih.gov/pubmed

baseados em abordagens de computação de dados intensivos (*Data Intensive Computing*), junto com tecnologias da Web Semântica para avaliar o processo de busca de fontes de dados relevantes, isto é fontes relacionadas a um domínio específico. Assim, podemos mitigar as limitações do estado de arte.

Este capítulo se baseia no processo de recomendação de fontes RDF descrito por Nikolov [15], o processo é composto de um conjunto de passos estruturados para identificar fontes relevantes. O objetivo desta abordagem é melhorar o processo de publicação, interligação e exploração de dados. Na seção 3.2.3 são discutidas algumas limitações referentes ao processo de Nikolov, que representam as motivações para o desenvolvimento deste trabalho.

O processo de recomendação de fontes RDF proposto em nosso trabalho, auxilia o usuário desde a análise do conjunto de dados que ele pretende publicar até a identificação de fontes candidatas, com as quais poderia conectar tópicos de interesse. O processo proposto é representado na Figura 4.1 a seguir.

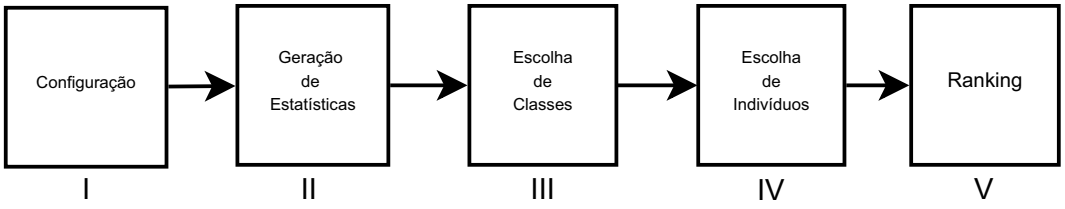

Figura 4.1: Processo de recomendação de fontes RDF

O processo é composto pelas etapas de configuração, geração de estatísticas, escolha de classes, escolha de indivíduos e ranking. Este processo é materializado em uma ferramenta baseada em componentes Web que apoia o usuário em um processo de publicação de dados. Estas etapas são descritas a seguir.

I. **Etapa de configuração**: Ajuda o usuário na configuração dos diferentes componentes que habilitam o processo de busca de fontes de dados RDF. Componentes como: credencias de acesso dos serviços na nuvem, *cluster*, fluxo de tarefas, armazenamento e instâncias.

II. **Geração de estatísticas**: Seguindo o processo definido por Nikolov [15], esta etapa auxilia o usuário na escolha de diferentes tipos de estatísticas, com o objetivo de coletar informações relevantes da fonte. Essas informações podem ser por exemplo, o número de triplas, lista de classes, número de predicados, número de propriedades, etc. As estatísticas são coletadas no formato sugerido pelo o vocabulário RDF VoiD [45].

III. **Escolha de classes**: Para coletar os indivíduos no processo de busca, o usuário precisa de uma etapa de escolha de classes para conseguir iden-

tificar os indivíduos representativos relacionados ao tópico de interesse, como também para restringir o espaço de busca, diminuindo o custo do processamento. Assim, mais informações da fonte são coletadas, incrementando o conhecimento sobre os dados que se pretende publicar (ou consumir).

IV. **Escolha de indivíduos**: Nesta etapa, uma busca de texto ajuda o usuário na identificação dos indivíduos. Esta etapa é apoiada pelas informações estatísticas coletadas nas etapas anteriores, possibilitando uma busca eficiente usando os indivíduos relevantes.

V. **Ranking**: Uma vez realizada a busca sobre o índice semântico, o usuário coleta informação de fontes RDF. Um ranking de fontes de dados é gerado e ordenado de forma decrescente pela quantidade de indivíduos coletados segundo os tópicos escolhidos pelo usuário.

Antes de detalhar o processo de recomendação de fontes RDF, no entanto, é importante destacar que existem duas visões de uso do mesmo:

Visão do responsável pela publicação: visão de quem normalmente conhece parcial ou totalmente os dados que deseja publicar ou conectar.

Visão do consumidor: visão de quem deseja consumir dados que considera relevantes para um domínio específico, ou começar uma análise sem ter nenhuma informação acerca da fonte.

A visão do responsável pela publicação dos dados é o foco deste trabalho é será utilizada para ilustrar o processo no restante deste capítulo.

Para ilustrar as etapas do processo proposto descritas nas seções subsequentes utilizaremos uma fonte RDF específica como exemplo. A fonte escolhida pertence à *The Open University*[4]. A fonte `data.open.ac.uk` disponibiliza vários repositórios de dados da universidade, repositórios relacionados a publicações, cursos, material de áudio e vídeo produzido na *The Open University*, assim como informações à respeito das pessoas envolvidas.

A *The Open University* atualmente desenvolve o projeto *LUCERO JISC Project*[5] que tem como foco extrair, conectar e disponibilizar dados entre diferentes repositórios da universidade. O objetivo principal deste projeto é compartilhar conhecimento educativo de aprendizagem a distância, usando os princípios de *Linked Data*, procurando conectar os dados com repositórios especificamente do setor educativo.

[4]http://www.open.ac.uk/
[5]http://lucero-project.info/lb/

Os dados disponibilizados por meio da fonte `data.open.ac.uk` se adequam aos fins da ilustração do processo proposto. Muitos desses dados ainda não se encontram totalmente conectados, oferecendo uma situação adequada para o processo de busca proposto. Do conjunto de dados publicados por *The Open University* selecionamos uma amostra para ilustrar o processo de recomendação de fontes RDF aqui proposto.

As próximas seções descrevem cada uma das etapas do processo proposto, seguindo a sequência do fluxo ilustrado na Figura 4.1.

4.1 Etapa de configuração

Antes de detalhar o processo proposto neste documento, é necessário conhecer os componentes da etapa de configuração, de forma a melhorar o entendimento do processo. Estes componentes habilitam o funcionamento das subsequentes etapas e são representados na Figura 4.2.

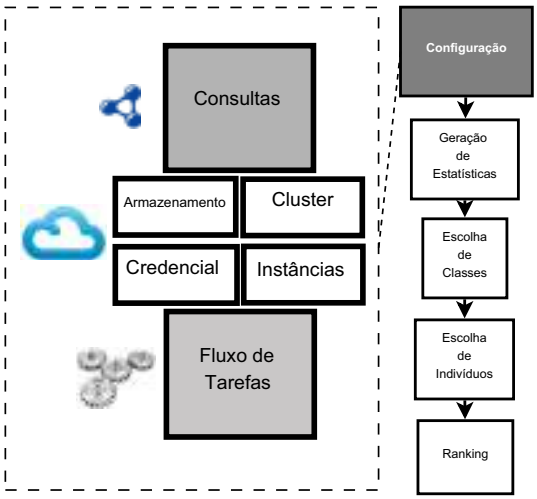

Figura 4.2: Detalhamento da etapa de configuração

Basicamente, a etapa de configuração é definida em três grupos, a seguir:

Configuração de serviços na nuvem: A informação que o usuário fornece nesta etapa influencia todas as configurações restantes. Identifica-se quatro tipos de informações:

1. Credencial: Define a identificação do usuário e permissões de acesso aos serviços na nuvem.
2. *Cluster*: Representa a descrição dos aspectos de *hardware* usado no processamento de dados, por exemplo, capacidade e quantidade de computadores.

3. Armazenamento: Define o diretório raiz onde os dados do usuário são carregados para serem processados, e também onde os resultados temporais e a estatística do processamento de dados são salvos.

4. Instâncias: Corresponde à descrição dos servidores responsáveis pelo armazenamento de dados. Definem-se dois tipos de servidores: um para armazenar os documentos RDF e outro para a indexação e a busca dos dados.

Gerenciamento do fluxo de tarefas: O usuário deve fornecer informações da configuração das tarefas MapReduce já definidas na arquitetura. As informações que definem um fluxo de tarefas, um *cluster* e opções de otimização são essências para a execução de um fluxo de tarefas MapReduce.

Gerenciamento de consultas SPARQL: O usuário define consultas em formato SPARQL [9], que serão executadas nas tarefas MapReduce. A informação recuperada ajuda a caracterizar o conteúdo da fonte de dados que o usuário pretende analisar, por exemplo, consultas do número de triplas, número de classes, número de indivíduos de uma classe, classes existentes na fonte, entre outras informações definidas pelo mesmo usuário.

Os detalhes destas configurações são apresentadas no capítulo 6.

4.2 Etapa de geração de estatísticas

Do ponto de vista do responsável pela publicação, esta etapa recupera as classes da fonte de dados escolhida, facilitando a escolha do tópico que se deseja publicar. Do ponto de vista do consumidor, esta etapa fornece um conjunto de consultas, previamente definidas, que recuperam informações estatísticas da fonte de dados. Esta etapa também fornece a possibilidade de retro-alimentar o conjunto de consultas oferecidas, otimizando as estatísticas obtidas. A etapa de geração de estatísticas é composta por três processos principais que agrupam o conjunto de operações representadas na Figura 4.3 a seguir.

O processo de entrada de dados propriamente ditos (seção 4.2.1) é supervisionado pelo usuário e agrupa três operações: (I) a escolha de fonte de dados, (II) a escolha de consultas SPARQL, que recuperam informações estatística da fonte e (III) a escolha do fluxo de tarefas que define o tipo de processamento de dados. O segundo processo é o processamento de dados (seção 4.2.2), supervisionado pelos serviços na nuvem e define uma sequência

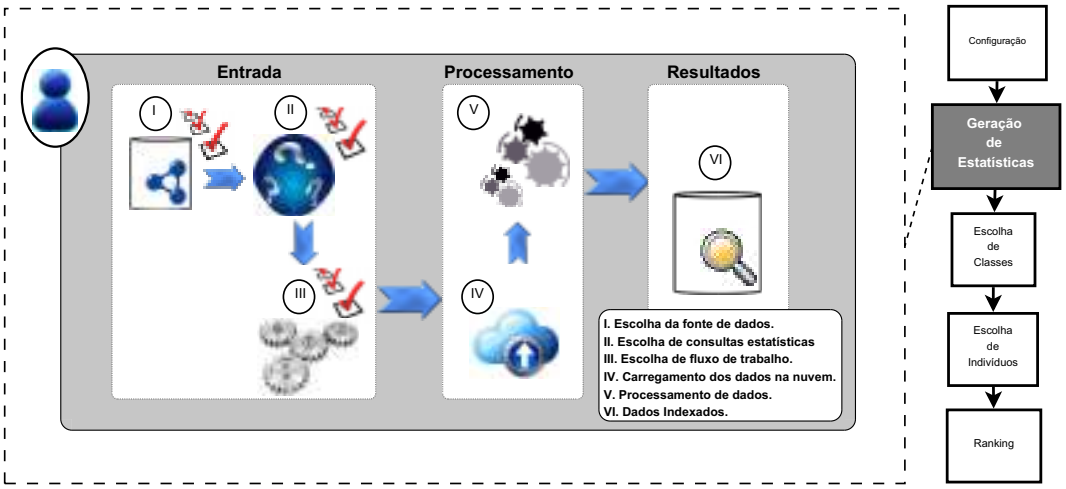

Figura 4.3: Detalhamento da etapa de geração de estatísticas

com duas operações: (IV) o carregamento de dados, onde os dados são transferidos do cliente até o serviço de armazenamento, e (V) o processamento realizado pelo serviço de MapReduce, sendo os dados armazenados e indexados. Por último, (VI) o resultado é disponibilizado ao usuário através de uma interface amigável (seção 4.2.3).

4.2.1 Entrada de dados

I. Escolha da fonte de dados

Os dados da Web semântica podem ser disponibilizados e consumidos de diferentes maneiras. Um banco de dados pode ser publicado como um simples arquivo RDF. Alternativamente, dentro de um banco de dados, a descrição de um recurso pode ser obtida dereferenciando a URI correspondente ao mesmo. Outros bancos de dados podem oferecer acesso aos dados por meio de um SPARQL *endpoint*, permitindo aos clientes enviar requisições usando o protocolo e a linguagem de consultas SPARQL RDF [25].

Embora por um lado estes métodos auxiliem na tarefa de disponibilização dos dados, eles podem ter consideráveis efeitos sobre a quantidade de banda de rede e recursos de computação consumidos, tanto do lado do cliente quanto do lado do servidor [25]. O processo que desenvolvemos permite o acesso aos dados via RDF *dump* e SPARQL *endpoints*.

RDF *dump*: Normalmente estes tipos de arquivos são oferecidos pelos fornecedores de dados, por exemplo, Geonames [6] oferece um arquivo

[6] http://www.geonames.org

dump completo de todo seu banco de dados, permitindo que esses dados possam ser diretamente importados e não extraídos iterativamente (crawling) [25].

Um arquivo *dump* RDF pode ser fornecido nos formatos RDF/XML, N-Triples e N-Quads. Opcionalmente os arquivos *dump* podem ser compactados em formatos GZIP, ZIP ou TAR [25]. Estes detalhes são abordados no processo proposto.

SPARQL *endpoint*: Embora esta seja a opção mais flexível para acessar dados RDF, o custo deste método de publicação é alto. Por exemplo, uma consulta precisa ser escrita de modo que respeite a sintaxe e semântica da linguagem SPARQL, precisa ser codificada, e o resultado traduzido em um formato útil para o usuário [25]. Assim, um processo que conta com um número excessivo de requisições, com execução de consultas complexas, pode tornar um banco de dados inoperante na web. Por esse motivo o responsável pela publicação limita o acesso aos dados [11].

O serviço SPARQL *endpoint* da DBpedia limita o acesso em termos de: (i) número de consultas por endereço IP (*Internet Protocol*) por segundo, (ii) tempo de execução de uma consulta, (iii) número de triplas retornadas.

O processo proposto apoiado na abordagem de computação na nuvem tenta lidar com estas limitações. Para o método SPARQL *endpoint*, é necessário configurar uma instância de busca e indexação de dados. Caso o método adotado seja o RDF *dump*, também é necessário configurar uma instância de armazenamento de dados RDF, que disponibiliza o arquivo *dump* através de um serviço de SPARQL *endpoint*. A configuração das instâncias deve ser feita na etapa de configuração, descrita na seção 4.1.

O método de acesso escolhido na fonte de dados `data.open.ac.uk`, de nossa ilustração é SPARQL *endpoint*, que tem como URL `http://data.open.ac.uk/query`.

II. Escolha de consultas SPARQL

Existem boas práticas de publicação de dados que os responsáveis dos dados devem seguir e, desta forma fornecer informação suficiente que auxilie os potenciais usuários a acessar e usar os dados [8]. Por exemplo, o vocabulário de interligação de conjunto de dados (VoiD - *Vocabulary of Interlinked Datasets*) consiste em um conjunto de instruções que possibilitam descobrir e usar um conjunto de dados conectados. VoiD é de interesse da W3C (*World Wide Web*

Consortium)[7], que incentiva o seu uso para descrição de conjuntos de dados em formato RDF [24].

O vocabulário VoiD é adotado na etapa de geração de estatísticas do processo, guiando a implementação de consultas SPARQL. As propriedades do VoiD são usadas como referência na definição de consultas que quantificam o conteúdo da fonte de dados. É importante destacar que essas implementações são parte do projeto *void-impl* [8], que motiva o uso do vocabulário VoiD.

A tabela 4.1 apresenta as relações entre as propriedades de VoiD e a suas respectivas consultas em SPARQL, implementadas pelo processo proposto de recomendação de fontes RDF.

Tabela 4.1: Relação entre as propriedades do VoiD e consultas SPARQL

ID	Propriedade	VoiD	SPARQL
Q1	Número de triplas	void:triples	SELECT (COUNT(*) AS ?no) {?s ?p ?o}
Q2	Número de classes	void:classes	SELECT COUNT(distinct ?o) AS ?no {?s rdf:type ?o }
Q3	Número de Entidades	void:entities	SELECT COUNT(distinct ?s) AS ?no {?s a [] }
Q4	Número de Sujeitos	void:distinctSubjects	SELECT (COUNT(DISTINCT ?s) AS ?no) {?s ?p ?o}
Q5	Número de Objetos	void:distinctObjects	SELECT (COUNT(DISTINCT ?o) AS ?no) { ?s ?p ?o filter(!isLiteral(?o))}
Q6	Número de propriedades	void:properties	SELECT COUNT(DISTINCT ?p) {?s ?p ?o}
Q7	Vocabulário: classes	void:vocabulary	SELECT DISTINCT ?type {?s a ?type}
Q8	Vocabulário: classes	void:vocabulary	SELECT DISTINCT ?type {?s a ?type} OFFSET **parametro:inicio** LIMIT **parametro:fator**

O usuário deve escolher as consultas que deseja executar. Além disso, existe a opção de adicionar outras consultas, que não estejam implementadas e que ajudem na coleta de informações estatísticas das fonte de dados consultadas.

Note na tabela 4.1, que as consultas **Q7** e **Q8** correspondem à propriedade `void:vocabulary` e recuperam as classes contidas na fonte de dados. Caso o número total de classes seja maior que o valor do **fator**, novas

[7] http://www.w3.org/

[8] http://code.google.com/p/void-impl/wiki/SPARQLQueriesForStatistics

consultas são criadas para recuperar todas as classes da fonte. A consulta **Q8** é parametrizada, reproduzida usando o valor do **fator** da fonte de dados. As propriedades *OFFSET* e *LIMIT* da linguagem SPARQL permitem realizar esta operação [9].

As consultas **Q7** e **Q8** são obrigatórias na etapa de geração de estatísticas.

III. Escolha do fluxo de tarefas para o processamento de dados

A definição de um fluxo de tarefas é composta por três componentes:

1. Uma sequência de execução de tarefas MapReduce.
2. Um *cluster*, onde é processado o fluxo de tarefas.
3. Diretórios de armazenamento de dados, onde devem ficar salvos os resultados e as estatísticas do processamento.

A figura 4.4 apresenta a definição de um fluxo de tarefas.

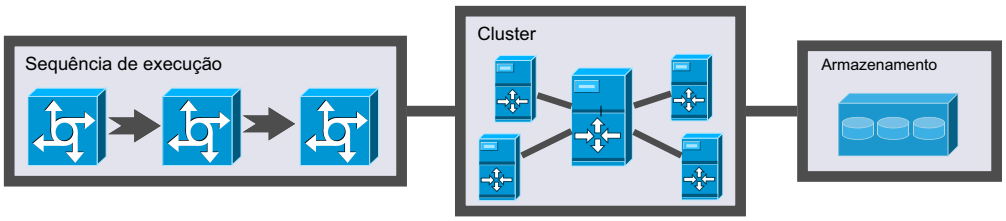

Figura 4.4: Fluxo de tarefas

Cada passo na sequência de execução deve ser definido na etapa de configuração. A estrutura de definição de um passo de execução contém:

Uma Entrada de dados: Diretório onde são carregados os dados para serem processados pelo serviço MapReduce. Este diretório é criado no serviço de armazenamento na nuvem.

Uma Saída de dados: Diretório onde são persistidos os resultados do processamento do serviço de MapReduce. Este diretório é criado também no serviço de armazenamento na nuvem.

O acesso SPARQL *endpoint* e o nome do grafo padrão (*Graph* IRI)[9] : Definem o acesso ao serviço que disponibiliza os dados da fonte escolhida.

[9] A definição de um conjunto de dados em SPARQL precisa de: *i.* Um nome, que é um IRI (*Internationalized Resource Identifier*) e *ii.* um grafo RDF. Um grafo padrão é usado por SPARQL quando o cliente não específica qual grafo usar. Estas definições formam a base da semântica das consultas SPARQL, assim cada consulta é executa sobre um conjunto de dados específico [9].

Para a fonte `data.open.ac.uk`, o valor de SPARQL *endpoint* é a url `http://data.open.ac.uk/query` e um valor vazio em *Graph IRI*. Se o método de acesso escolhido for RDF *dump*, estes valores são gerados automaticamente, e informados ao usuário.

Fator: Este valor representa a quantidade máxima de resultados que o serviço de SPARQL *endpoint* pode retornar. Por exemplo, o serviço `http://data.open.ac.uk/query` da fonte escolhida pode retornar no máximo 200 resultados.

Serviço de indexação: Define o acesso ao serviço de indexação de dados, onde ficam salvos os resultados finais do processamento. Este serviço é criado automaticamente sobre a instância de indexação e a busca de dados. A instância é definida na etapa de configuração descrita na seção 4.1.

Opções de otimização: Definem como distribuir o volume do processamento do passo de execução entre os processos MapReduce. Os detalhes destes valores são detalhados no capítulo 6.

A escolha do fluxo de tarefas tem relação direta com a disponibilidade de recursos na nuvem e da informação previa que o usuário possui acerca da fonte de dados. Com essa informação o usuário pode ter uma idéia clara da possível quantidade do processamento que o serviço MapReduce executará.

Na sequência de execução de tarefas MapReduce são executadas as consultas indicadas pelo usuário. Os detalhes da implementação das tarefas MapReduce são descritos na seção 6.2 do capítulo 6.

4.2.2 Processamento

IV. Carga de dados

Após a entrada de dados ser definida pelo usuário, é necessário carregar os dados no serviço de armazenamento para que possam posteriormente ser consumidos pelo serviço MapReduce. Antes de continuar com a ilustração, um resumo dos dados carregados no serviço de armazenamento é apresentado na tabela 4.2, considerando a fonte de dados `data.open.ac.uk`.

No resumo da tabela 4.2 é definido um *cluster* de 1 computador. O critério de definição do *cluster* é baseado no número de consultas que se pretende executar e no valor de **fator** da fonte de dados. Recomenda-se executar a consulta **Q2** no serviço de SPARQL *endpoint* para ter uma idéia do número

Tabela 4.2: Resumo de entrada de dados

Parâmetro	**Valor**
Fonte de dados	
Nome	`data.open.ac.uk`
Método de acesso	SPARQL *endpoint*
Fluxo de tarefas	
Entrada	diretorioraiz/entrada_{data}
Saída	diretorioraiz/saida_{data}
SPARQL *endpoint*	`http://data.open.ac.uk/query`
GIRI	VAZIO
Fator	200
Serviço de Indexação	nome_instancias_gerada
Cluster	
Número de computadores	1
Tipo de computador	m1.small[10]
Memoria RAM	1.7 GB
Armazenamento	160 GB
Consultas	
Lista de Consultas	{Q1, Q2, Q3, Q4, Q5, Q6, Q7, Q8}

de classes que se pode conseguir. Por exemplo, no banco de dados da DBPedia há um total de 288529 classes e um fator de 10000 resultados por consulta [46]. Para recuperar todas as classes do banco são necessárias 29 consultas extras do tipo **Q8**, resultando em um total de 36 consultas a serem executadas. Segundo os testes realizados no processo apresentado, a DBPedia precisa de um *cluster* de 1 computador, no mínimo, para recuperar todas as classes do banco de dados.

No momento da elaboração deste trabalho, a fonte de dados `data.open.ac.uk` continha 68 classes, com um **fator** de 200. Consequentemente, são necessárias somente 8 consultas executadas em um *cluster* de 1 computador para atender esta demanda de processamento.

V. Processamento de dados

A computação na nuvem para o processamento dos dados tem como requisitos as seguintes características [17]:

Cache distribuída: Que permite mover os dados até o computador que realiza o processo computacional.

Processamento distribuído: Que ajuda a mover o aplicativo que executa a computação dos dados até o computador onde estes estão hospedados.

[10] **m1.small** é um tipo de computador definido em *Amazon Web Services* [39]

Estas características são fundamentais na computação intensiva de dados (aplicações do tipo *I/O bound*), pois dedicam uma fração do tempo de execução para mover dados. Fatores como sobrecarga do protocolo de rede, latência da rede, largura da banda de interconexão e de computação, também devem ser considerados neste tipo de processamento [17]. No processo proposto, estes fatores são assumidos pelo serviço de MapReduce, mas os tempos de resposta dos serviços de SPARQL *endpoint* e a indexação de dados não são considerados pelo serviço. Estes últimos fatores são assumidos pelo processo proposto e são detalhados no capítulo 6.

Cada uma das consultas SPARQL é carregada no serviço de armazenamento e atribuída à uma tarefa MapReduce. Uma consulta é executada e monitorada por uma API (do inglês *Application Programming Interface* ou interface para programação de aplicações) SPARQL e o resultado das consultas é consumido por outra API, a qual o direciona ao serviço de indexação. O resultado é um conjunto de classes, onde cada classe é armazenada como um documento independente no serviço de indexação. A estrutura do documento gerado na coleta permite persistir informação da classe recuperada. Um exemplo da estrutura do documento indexado é apresentada no apêndice B, na tabela B.1.

4.2.3 Resultados

Depois que o processamento de dados da etapa de geração de estatística finaliza, os dados são consumidos pelo serviço de indexação e eles são disponibilizados para o usuário. Na tabela 4.3 é apresentado o resultado da execução das consultas da tabela 4.1. O número de classes obtido, é menor se comparado ao DBPedia. Este resultado é relevante, se for levado em conta à quantidade de informação contida na fonte de dados.

Tabela 4.3: Resultado de consultas

ID Consulta	Nome	Valor
Q1	Número de triplas	2812085
Q2	Número de classes	68
Q3	Número de entidades	227473
Q4	Número de sujeitos	410750
Q5	Número de objetos	340618
Q6	Número de propriedades	441

Outros resultados desta etapa são apresentados na seção 5.1, no capítulo 5. Estes resultados representam a saída das consultas indicadas pelo usuário.

4.3 Etapa de escolha de classes

A próxima etapa do processo proposto consiste na escolha de um conjunto de classes relacionado ao tópico de interesse do usuário, para assim, coletar os indivíduos que representam as classes e iniciar o processo de busca.

Nesta etapa, para aperfeiçoar o processo de busca algumas questões devem ser levadas em conta:

(a) Como escolher uma classe?

(b) Que informação de um indivíduo é necessário coletar?

(c) Que fatores influenciam o processo de busca de classes?

A etapa de escolha de classes é composta por um conjunto de operações. Similar à etapa de geração de estatísticas, descrita na seção 4.2, as operações da etapa de escolha de classes também são agrupadas em três processos principais: entrada de dados, Processamento de dados e Resultados obtidos. Estes processos são representados na Figura 4.5.

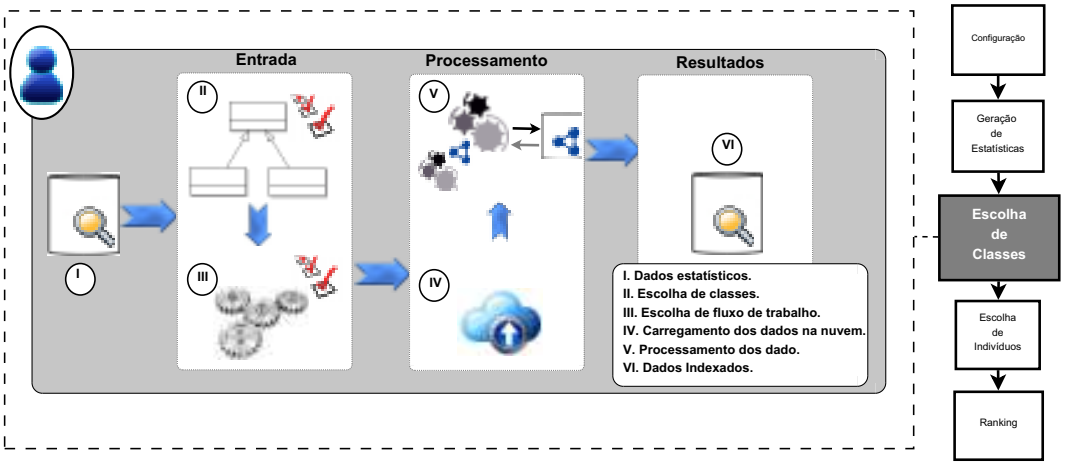

Figura 4.5: Detalhamento da etapa de escolha de classes

Nesta etapa os dados são fornecidos pelo serviço de indexação. O processo de entrada de dados contém duas operações (seção 4.3.1): (II) a escolha de classes e (III) a escolha de um fluxo de tarefas para o processamento dos dados. As operações do processo de entrada são supervisionadas pelo usuário. O processamento de dados (seção 4.3.2), que é resolvido pela arquitetura na nuvem, é composto por: (IV) uma carrega de dados e (V) o processamento dos dados através do serviço MapReduce. Por último, o (VI) resultado desta etapa é armazenado, indexado e disponibilizado ao usuário (seção 4.3.3).

4.3.1 Entrada de dados

II. Escolha de classes

Após a geração das estatísticas, o usuário tem informação suficiente para escolher as classes que representam o seu domínio. Porém, em uma fonte de dados como a DBpedia, com 288529 classes [46], a escolha de uma classe não é trivial. Esta tarefa torna-se ainda mais complexa para um usuário com pouca experiência em SPARQL.

Existem técnicas de processamento de texto que facilitam a extração automática de relações e conceitos. Algumas delas ajudam a caraterizar e classificar o texto que pretende-se analisar [47]. É importante considerar que parte do texto analisado pertence a URIs. O serviço de indexação habilita técnicas que são usadas na busca de classes. A seguir são apresentados exemplos destas técnicas:

- **Busca de padrões de caracteres**, para o padrão ***acad*** um dos resultados recuperados é:

  `http://purl.org/ontology/bibo/`**`AcademicArticle`**.

- **Toquenização de dados:** Uma toquenização por palavra da URI anterior é:

  URI: `http://purl.org/ontology/bibo/AcademicArticle`

  Toquens: {http, purl, org, ontology, bibo, academic, article}

  Percebe-se a toquenização por transição de letra minuscula para maiúscula, o termo "AcademicArticle" é dividido em "academic" e "article".

  Toquenizações hierárquicas são também consideradas na indexação como por exemplo:

  Toquens: { http://purl.org/ontology/bibo/AcademicArticle,

  http://purl.org/ontology/bibo/, http://purl.org/ontology/,

  http://purl.org/, http://}.

- **Filtro de palavras:** Palavras comuns são filtradas, por exemplo:

  Palavra filtrada: {"a", "an", "and", "be", "but", "by", "if", "in", "not", . . .,"etc"}.

  Podemos definir também o tamanho das palavras que serão filtradas.

– **Índice invertido:** O índice invertido também é chamado de arquivo invertido. Ele armazena pares de chaves valores (w, L), onde w é uma palavra e L é uma coleção de documentos que contêm a palavra w.

Por exemplo, a lista de prefixos recuperadas da fonte `data.open.ac.uk`, na tabela 5.2, é recuperada usando esta técnica. O índice criado considera a URI da classe como documento e cada prefixo como palavra.

Índice$(w, |L|)$: {(`http://purl.org`, 24), (`http://data.open.ac.uk`, 18), (`http://www.w3.org`, 11),..., (`http://dbpedia.org`, 1),...}

No processo de busca de classes, essas técnicas facilitam a escolha e ajudam o usuário na identificação dos tópicos de interesse. Este tipo de busca é baseado em palavras-chave, que é um dos métodos mais usados para acessar informação na Web. No entanto, o modelo palavra-chave na Web de Dados não é muito apropriado. Isto porque os documentos da Web de Dados não são só um conjunto de palavras, eles contêm dados semiestruturados [48]. Portanto, sistemas tradicionais de recuperação de informação baseados em texto não tem os mesmos resultados, eles são incapazes de capturar a informação estrutural dos documentos [48]. Esta problemática é discutida neste capítulo, na seção 4.4.

Para a ilustração do processo foram escolhidas 3 classes da fonte `http://data.open.ac.uk/`. A classe **bibo:***AcademicArticle* representa documentos acadêmicos e é uma especialização de **foaf:***Document*, com um total de 16013 documentos. As classes `http://data.open.ac.uk/podcast/ontology/VideoPodcast` e `http://data.open.ac.uk/podcast/ontology/AudioPodcast` representam os recursos de multimídia dos *courseware*[11] da universidade, com um total de 3788 arquivos.

III. Escolha de fluxo de tarefas

Como na etapa de geração de estatísticas descrita na seção 4.2, define-se a escolha de um fluxo de tarefas. A definição deste fluxo tem relação direta com a quantidade de dados que se pretende processar. Uma bom parâmetro para medir o volume de processamento é o número total de indivíduos por classe.

O processamento da sequência de tarefas MapReduce é feito em dois passos básicos: o primeiro define a quantidade de indivíduos por classe, e o segundo permite extrair os indivíduos destas classes. Existem outras

[11] *Courseware*é um software para uso educacional

considerações que são detalhadas na seção 6.2, no capítulo 6, acerca da definição do fluxo de tarefas MapReduce da escolha de classes.

4.3.2 Processamento

IV. Carga de dados

Como na etapa de geração de estatísticas, a entrada de dados é carregada no serviço de armazenamento e consumida pelo serviço MapReduce. O resumo dos dados carregados é apresentado na tabela 4.4, onde lista-se o conjunto de classes indicadas pelo usuário e as consultas necessárias para a extração das informações. A configuração do processamento de dados da etapa da escolha de classes é apresentada na tabela 4.5.

Tabela 4.4: Lista de classes e consultas auxiliares

ID	Nome	Valor
Lista de classes		
C1	Artigos Acadêmicos	**bibo:***AcademicArticle*
C2	Documento de áudio	`http://data.open.ac.uk/podcast/ontology/VideoPodcast`
C2	Documento de vídeo	`http://data.open.ac.uk/podcast/ontology/AudioPodcast`
Lista de consultas		
Q9	Número de indivíduos por classe	SELECT (COUNT(DISTINCT ?i) AS ?c){?i a **parametro:clase**}
Q10	Indivíduos por classe	SELECT DISTINCT ?i WHERE{?i a class} OFFSET **parametro:inicio** LIMIT **parametro:fator**}
Q11	Propriedades do indivíduo	SELECT DISTINCT ?p ?o WHERE{**parametro:individuo** ?p ?o}
Q12	Relações do indivíduo	SELECT DISTINCT ?s ?p WHERE {?s ?p **parametro:individuo**}

Além das classes escolhidas, é mandatório encaminhar uma lista de consultas, as quais recuperam informação relevante dos indivíduos. A consulta **Q9** recupera a quantidade de indivíduos referente às classes atribuídas no parâmetro exigido. A consulta **Q10** recupera os indivíduos usando o parâmetro **fator**, que limita a quantidade de resultados que a fonte de dados permite recuperar por consulta. As consultas **Q11** e **Q12** recuperam informação do indivíduo segundo sua posição na tripla, assim podemos obter informação das propriedades que contêm e das relação onde participam.

Tabela 4.5: Configuração do processo

Parâmetro	**Valor**
Fonte de dados	
Nome	`data.open.ac.uk`
Método de acesso	SPARQL *endpoint*
Fluxo de tarefas	
entrada	diretorioraiz/entrada_{data}
saída	diretorioraiz/saida_{data}
SPARQL *endpoint*	`http://data.open.ac.uk/query`
GIRI	VAZIO
Fator	200
Serviço de Indexação	nome_instancias_gerada
Cluster	
Número de computadores	15
Tipo de computador	m1.xlarge[12]
Memória RAM	15 GB
Armazenamento	1690 GB

É importante ressaltar que cada uma destas consultas podem ser especializada. Assim, restringimos o número de indivíduo recuperados por classe. Por exemplo, podemos agregar na consulta alguma determinada propriedade que todo o indivíduo deve conter, ou, filtrar algum indivíduo que pertença a uma classe específica.

Diferente do processamento da etapa de geração de estatísticas, esta etapa exige de um grande numero de computadores no *cluster*. Porém, as opções de otimização do fluxo de tarefas na recuperação de indivíduos podem variar, para assim, balancear o volume de processamento entre os computadores do *cluster*. As opções de otimização são descritas na seção 6.3, no capítulo 6.

V. Processamento de dados

No processo proposto, a abordagem de recuperação de informação está baseada em entidades. Este tipo de busca é nomeado como "Busca centrada em entidades" (do inglês *Entity centric search*), onde uma entidade é um recurso cuja descrição é reunida de diferentes fontes [11, 48]. Na etapa da escolha de indivíduos na seção 4.4, apresenta-se os detalhes desta abordagem. O processo proposto consegue reunir informações destes recursos, que de forma geral são nomeados a partir de indivíduos, por pertencer a classes específicas que se pretende relacionar. A informação dos indivíduos recuperada

[12] **m1.xlarge** é um tipo de computador definido em *Amazon Web Services* [39]

é armazenada e indexada como um documento. O documento considera a natureza semiestruturada da informação.

O processo de busca reúne dois tipos de informação para cada indivíduo, escritos a seguir:

1. **Propriedades e Relações do indivíduo:** Estas informações são recuperadas usando as consultas **Q11** e **Q12**, que são descritas na tabela 4.1.

2. **Informação da dereferenciação da URI do indivíduo:** A URI de cada indivíduo é dereferenciada utilizando-se o protocolo HTTP, o dereferenciamento de uma URI resulta em uma descrição RDF do recurso identificado, com objetivo de recuperar mais informação [1, 49]. Este processo considera que nem todos os dados são publicados no serviço de SPARQL *endpoint* [2]. Esta informação complementa os dados obtidos através do RDF *dump* que disponibiliza subconjuntos de dados da fonte.

Estas informações são recuperadas através de uma operação *merge*. A operação *merge* executa as consultas **Q11** e **Q12**, da tabela 4.1, junto com o processo de dereferenciação. As informações são analisadas para eliminar as possíveis redundâncias. Estas informações são caracterizadas e materializadas em um documento. Na Figura 4.6,apresenta-se uma ilustração da operação de *merge*. O documento gerado é consumido pelo serviço de indexação. A estrutura do documento gerado é apresentado no apêndice B, na tabela B.3.

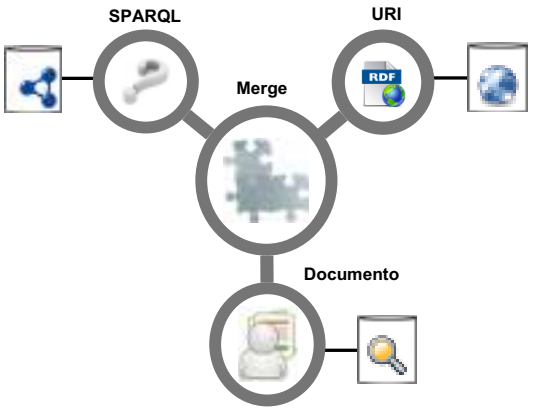

Figura 4.6: Merge de informação do indivíduo

No processamento de dados, a operação de *merge* é atribuída a uma tarefa MapReduce. Uma operação *merge* recupera a informação somente de um indivíduo. No caso de classe **bibo:***AcademicArticle* o número de indivíduos é 16013, essa mesma quantidade é equivalente ao número operações *merge* necessárias para recuperar as informações dos indivíduos.

Um fator importante para o sucesso da operação de *merge* é a coordenação do tempo de resposta de cada um dos serviços usados no processamento de dados. Por exemplo, se a consulta ao serviço de dados SPARQL falhar, é ativada uma espera de uma determinada quantidade de tempo, até ser executada novamente. No serviço de indexação, a concorrência de requisições de indexação é um problema resolvido da mesma forma. O valor de espera de tempo das requisições e as quantidades de tentativas são opções de otimização configuráveis do processo proposto.

A análise da descrição do erro, em caso de falha nas requisições, ajuda a definir uma ação no fluxo de processamento. Possíveis falhas do serviço, ausência do recurso solicitado, e qualidade dos dados, afetam o processamento das tarefas MapReduce.

4.3.3 Resultados

Para a classe de artigos acadêmicos da fonte `data.open.ac.uk` foi possível coletar documentos de 16003 indivíduos, com uma perda de 10 documentos. Esta margem de erro é resultado dos tempos de resposta do serviço de indexação. Comprovou-se a existência dos indivíduos solicitados, mas o serviço de indexação não conseguiu processar os respectivos documentos.

Para as classes que representam o material multimídia foram coletados documentos de 3788 indivíduos, onde 2240 são de vídeo e 1548 de áudio. Não ocorreu perda de dados nesta coleta.

Estes resultados foram obtidos usando a configuração apresentada na tabela 4.5.

Os resultados desta etapa são detalhados na seção 5.2, do capítulo 5. Nesta seção, também são apresentadas diferentes estatísticas sobre as indivíduos das classes escolhidas, e que são de grande utilidade para o usuário no momento da escolha de indivíduos relevantes.

4.4 Etapa de escolha de Indivíduos

Nesta etapa identificam-se os indivíduos que fazem parte da busca de fontes, os quais representam os tópicos de interesse que pretende-se conectar. Para isto, o usuário precisa resolver uma questão:

> *Como identificar um indivíduo se não há certeza do conteúdo exato que ele possui?*

É claro que a linguagem padrão para consultas de dados em formato RDF é a linguagem SPARQL, onde uma consulta está composta de conjunções

de elementos de busca tipo <*Sujeito*> <*Predicado*> <*Objeto*>, que recebe o nome de **grafo padrão** [50]. No entanto, a linguagem é limitada quando se trata de procurar dados em coleções muito heterogêneas. A variação na estrutura e nos vocabulário dos dados torna difícil a escrita de uma consulta SPARQL [43].

Existem outros estudos sobre buscas baseadas em palavra-chave [51, 52, 53, 54], para dados na Web semântica e dados RDF. Estes trabalhos aproveitam a simplicidade da busca por palavra-chave, dado que este tipo busca é fácil de usar pelo usuário, pois ela esconde qualquer informação estrutural subjacente à coleção de dados. Mas estas abordagens não aproveitam a natureza estruturada que oferece os documentos descritos em RDF [43].

Por outro lado, abordagens [55, 56, 57, 58, 59] baseadas em palavras-chave tem limitação ao expressar vários graus da estrutura dos dados, quando o usuário tem conhecimento parcial da fonte. Estes trabalhos estendem a busca por palavra-chave com capacidade de consultas estruturadas, assim, agregam operadores sobre os atributos e valores dos dados.

O aumento de expressividade das consultas tem relação direta com a complexidade de processamento, e muitos dos modelos baseados em grafos [55, 56, 57] não são aplicáveis em grande escala.

O processo proposto adota o modelo definido por Delbru [43], baseado em recuperação de entidades com informação de dados semiestruturados. O trabalho descrito por Delbru é focado na indexação de dados, mas o modelo pode ser usado para a análises de *links*, e abordagens de inferência distribuída. O autor apresenta o Motor de Recuperação de Informação Semântica (do inglês *Semantic Information Retrieval Engine*, SIREn), usado na identificação de indivíduos relevantes, para eles serem usados na busca de fontes de dados. Uma referência detalhada de como usar este motor pode ser encontrada em [60].

A etapa de escolha de indivíduos, detalha a seguir, consiste de operações executadas em sequencia, como ilustrado na figura 4.7. São elas entrada de dados, processamento de dados e resultados.

O fluxo de operações consome dados estatísticos geradas nas etapas anteriores. Como na etapa de escolha de classes, esta etapa fornece ao usuário uma escolha dos dados como parte da entrada de dados para serem processados (seção 4.4.1). Nesta escolha são definidos os indivíduos das classes recuperadas e um fluxo de tarefas. O carregamento e o processamento de dados são tratados pelos serviços de nuvem (seção 4.4.2). Finalmente, os resultados são armazenados no serviço de indexação (seção 4.4.3). Estes resultados correspondem com as fontes encontradas.

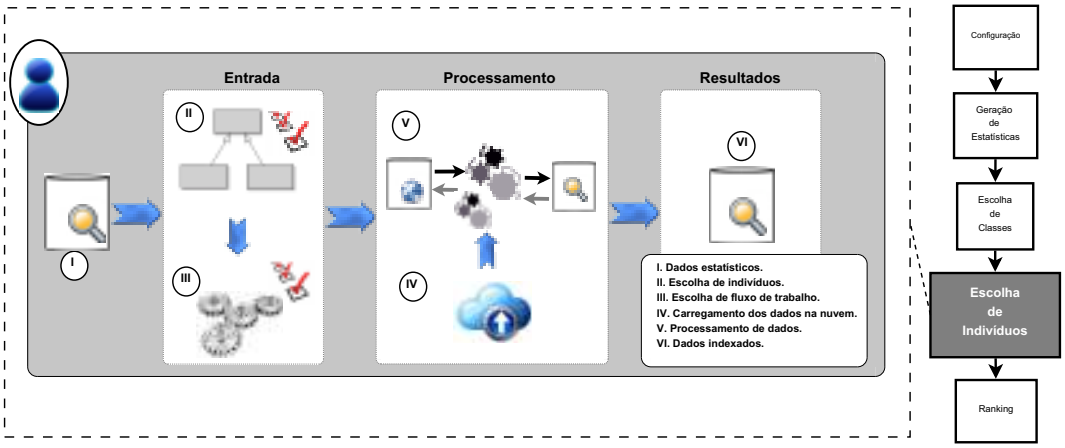

Figura 4.7: Detalhamento da etapa de escolha de indivíduos

4.4.1 Entrada de dados

II. Escolha de Indivíduos

O número de propriedades de tipo de dados (do inglês *data properties*) é considerado como fator fundamental na escolha de indivíduos. Segundo as estatísticas das tabelas 5.4 e 5.6 do capítulo 5, que apresentam uma distribuição da quantidade de relações e propriedades por documento das classes escolhidas, os documentos correspondentes aos indivíduos que contêm um número importante de propriedades são relevantes no processo de busca.

Deve-se analisar também o tipo de informação dos valores que contêm as propriedades. No caso da classe de "Artigo Acadêmico", existem três propriedades que colaboram com informação de tipo texto na descrição dos dados, estas propriedade são **rdfs:***label*, **bibo:***abstract* e **dc:***title*. As outras propriedades não são relevantes na extração de palavras-chave para o processo de busca, como é recomendado por Nikolov [15]. As propriedades com valores de tipo **data** ou **número** não são consideradas na busca, e são filtradas automaticamente no processo de extração.

Por outro lado, informação sobre a propriedade mais usada no documento coletado também ajuda no filtro de indivíduos. Por exemplo, nos documentos da classe Artigos Acadêmicos observa-se que a propriedade **bibo:***uri* de tipo texto é bastante usada na descrição dos indivíduos, como apresenta a tabela 4.6 (estás informações são obtidas na etapa de escolha de classes, na seção 4.5). Consequentemente nas estatísticas da tabela 5.4, o número de propriedades de tipo de dados apresenta um valor considerável para a descrição de um só documento, isto como resultado do uso frequente da propriedade **bibo:***uri*. Infelizmente o valor desta propriedade é de natureza URI, então os indivíduos

que usam esta propriedade não são relevantes na busca de fontes de dados.

Portanto, para identificar indivíduos que colaboram significativamente no processo de busca, é necessário encontrar indivíduos com o maior número de propriedades de tipo texto e filtrar indivíduos com propriedades que não são relevantes na busca de fonte de dados.

Estas situações oferecem um cenário conveniente para testar as capacidades da abordagem de busca por palavra-chave proposto por Delbru [43]. Para realizar uma busca na coleção de dados é necessário criar uma consulta segundo a sintaxe apresentada por McCandless [60]. O principal artefato de SIREn é o caractere genérico ***** (do inglês *wildcard*), para consultar múltiplos campos da estrutura do documento. SIREn também fornece um conjunto de operadores para executar operações sobre o conteúdo ou sobre a estrutura das tuplas do documento. Operadores booleanos, operadores de prefixos, operadores de proximidade, e simples buscas de termos ou frases, ajudam a identificar a informação desejada [60].

Na lista 4.1, apresenta-se a consulta que recupera os indivíduos que não contêm a propriedade de dados `http://purl.org/ontology/bibo/uri` (linha 1), e aqueles que contêm triplas com a propriedade **rdfs:***label* (linha 1). O resultado é ordenado de forma decrescente pelo campo **numprodado** (linha 2) e apresenta o campo **triple** do documento (linha 3).

Lista 4.1: Seleção de indivíduos da classe artigo acadêmico

```
1  nq=(* <label> *)NOT(* <uri> *) &
2  sort=numprodado desc&
3  field=triple
```

A consulta da lista 4.1 é encaminhada como uma requisição HTTP ao serviço de indexação, o resultado da consulta é formatado e apresentado na interface de usuário. Ressalta-se a simplicidade da sintaxe da consulta com operadores geralmente conhecidos, ao contrário da complexidade que fornece uma consulta SPARQL [60]. Este tipo de busca é mais adequado quando o usuário conhece os dados que estão contidos na fonte de dados.

A consulta recupera são um total de 9675 indivíduos, contendo os documentos dos indivíduos correspondentes 14 propriedades de tipo de dado como máximo e como mínimo 1. Este número de documentos é suficiente para começar a definir uma quantidade de adequada para a busca de fontes.

Para a coleção de material multimídia aplicam-se os mesmo critérios, mas a distribuição das propriedades dos indivíduos nos documentos coletados é mais

Tabela 4.6: Propriedades da classe artigo acadêmico

ID	Propriedade	Tipo	Número de documentos
Propriedade de tipo de dado (*Data Properties*)			
PD1	http://purl.org/dc/terms/date	date	16002
PD2	http://purl.org/dc/terms/title	string	15935
PD3	http://www.w3.org/2000/01/ rdf-schema#label	string	15935
PD4	http://purl.org/ontology/bibo/abstract	string	13803
PD5	http://purl.org/ontology/bibo/volume	string (integer)	11407
PD6	http://purl.org/ontology/bibo/issue	string (integer)	10278
PD7	http://purl.org/ontology/bibo/uri	string (URI)	6281
Propriedade de objeto (*Object Properties*)			
PO1	http://www.w3.org/1999/02/ 22-rdf-syntax-ns#type	URI	16003
PO2	http://purl.org/dc/terms/creator	URI	16003
PO3	http://purl.org/ontology/bibo/ authorlist	URI	16003
PO4	http://purl.org/dc/terms/ispartof	URI	16003
PO5	http://purl.org/ontology/bibo/status	URI	15996
PO6	http://purl.org/ontology/bibo/ presentedat	URI	14415
PO7	http://www.w3.org/2002/07/ owl#sameas	URI	2254
PO8	http://purl.org/dc/terms/publisher	URI	452
PO9	http://purl.org/ontology/bibo/ editorlist	URI	130

uniforme, como é apresentada na tabela 5.6 da seção 5.2, do capítulo 5. A tabela A.1 do apêndice A apresenta as propriedades das classes de material multimídia e como são distribuídas na coleção de documentos (estás informações são obtidas na etapa de escolha de classes, na seção 4.5). Considerando esta informação, a decisão de filtrar os indivíduos com a propriedade **bibo:***uri* não é adequada.

No entanto, o material multimídia está relacionado com diferentes tópicos de interesse como: ciência, saúde, tecnologia e engenharia, filosofia, literatura, entre outros, um total de 114 tópicos de interesse. Isto pode resultar em uma busca muito diversa, portanto, os indivíduos desta coleção são filtrados somente por um tópico. O tópico escolhido é `http://data.open.ac.uk/topic/engineering_and_technology` e é usado na descrição de 307 indivíduos. A consulta usada para identificar os indivíduos é apresentada a seguir.

Lista 4.2: Seleção de indivíduos das classes de material multimídia

```
1  nq=*+<subject>+<engineering>&
2  sort=numprodado desc&
3  field=triple
```

Na lista 4.2 a consulta apresentada filtra os indivíduos usando o predicado `http://purl.org/dc/terms/subject` (linha 1), tendo como objeto o valor de "engineering", termo que forma parte da URI `http://data.open.ac.uk/topic/engineering_and_technology`. A consulta ordena o resultado pelo campo **numprodado** (linha 2) e apresenta o campo **triple** do documento (linha 3). Uma vantagem da abordagem apresentada por McCandless [60], é considerar o *namespace* dos vocabulários usados, assim a estrutura da consulta é definida com uma complexidade baixa e fácil de entender. Por outro lado, os termos das buscas não precisam ser exatos para conseguir fazer uma busca, é claro que o espaço de resultados pode variar de acordo com o número das possíveis ocorrências encontradas, ou seja, se existem dois vocabulários diferentes com o mesmo nome do recurso, o motor de busca retorna os indivíduos de ambos vocabulários.

III. Escolha de fluxo de tarefas

Como em todas as etapas é necessário definir um fluxo de tarefas. A definição deste fluxo tem relação direta com a quantidade de dados que se pretende processar. Uma referência básica do volume de processamento é o número de indivíduos escolhidos. Outra referência é a quantidade de palavras-chave por documento que é gerada para executar a busca de fontes de dados. Este último parâmetro deve ser ajustado na etapa de configuração, descrita na seção 4.1.

O fluxo de tarefas MapReduce é composto por dois passos: o primeiro extrai as palavras-chave dos documentos que representam os indivíduos coletados, e o segundo envia as buscas por palavra-chave para o índice semântico. Os detalhes deste fluxo são detalhados na seção 6.2, do capítulo 6.

4.4.2 Processamento

IV. Carregamento de dados

Depois que o usuário realiza a escolha de indivíduos, gera-se uma lista destes indivíduos que posteriormente é carregada no serviço de armazenamento. Esta lista é carregada juntamente com as configurações do fluxo de tarefas. Um resumo dos dados carregados é apresentado na tabela 4.7.

Tabela 4.7: Configuração do processo

Parâmetro	**Valor**
Lista de Indivíduos	
Lista	url1,url2, url3, . . . urln
Fluxo de tarefas	
entrada	diretorioraiz/entrada_{data}
saída	diretorioraiz/saida_{data}
termos	10
Serviço de Indexação	nome_instancias_gerada
Cluster	
Número de computadores	5
Tipo de computador	m1.xlarge[13]
Memória RAM	15 GB
Armazenamento	1690 GB

V. Processamento de dados

Identificados os indivíduos, é necessário extrair o conjunto de palavras-chave dos documentos. Estas palavras-chave são encaminhados ao índice semântico na Web para realizar a busca de fontes e este índice contém uma amostra significativa do que é publicado na *Linked Open Data* e na Web. Assim, como é indicado no processo por Nikolov [15], o índice semântico recebe a requisição e responde com um conjunto de indivíduos que pertencem a uma determinada fonte de dados. Para realizar a busca é escolhido o índice semântico Sindice (do inglês *The Semantic Web Index*). O Sindice é composto por um conjunto de serviços web *on-line* que disponibiliza os dados coletados das diferentes fontes de LOD. Para enviar uma requisição ao Sindice é usada uma API, que realiza o controle da requisição e dos resultados retornados pelo índice. Detalhes da arquitetura do Sindice podem ser encontrados em [12].

Na extração de palavras-chave são usados conceitos de modelo de recuperação de informação, onde um documento é representado por um conjunto de palavras-chave, que recebem o nome de **termos de índice** [61]. O **termo de índice** é uma palavra cuja semântica representa o conteúdo de um documento. Para cada termo, pode-se atribuir um peso, isto é, um valor numérico que indicará o grau de relevância daquele termo ao descrever o conteúdo semântico de um documento [61].

O modelo vetorial definido por Baeza-Yates [61], representa os **termos de índice** dos documentos e das consultas como um espeço vetorial. Assim, os valores atribuídos aos termos são usados para calcular o grau de similaridade entre os documentos armazenados no sistema e a consulta do usuário.

[13] **m1.xlarge** é um tipo de computador definido em *Amazon Web Services* [39]

No modelo vetorial são usados três fatores [61] para este cálculo:

1. Frequência do termo no documento (**TF** do inglês *Document Term Frequency*): O **TF** calcula quantas vezes o termo t_i aparece no documento d_i .

2. Frequência do termo entre os documentos (**DF** do inglês *Document Frequency*): O DF mede a frequência do termo t_i na coleção de documentos.

3. Inverso da frequência do termo entre os documentos (**IDF** do inglês *Inverse Document Frequency*): O IDF mede o inverso da frequência do termo t_i na coleção de documentos.

O fator IDF mede a importância de uma palavra, ou seja, se uma palavra aparece com frequência nos documentos, ela não pode ser usada para distinguir objetos da coleção. Estes fatores são calculados usando as seguintes equações a seguir:

$tf_{i,j}$: Frequência de um termo t_i no documento d_j .

$f_{i,j}$: Frequência normalizada de um termo t_i no documento d_j .

$\max_l f_{l,j}$: Frequência máxima de todos termos no documento d_j .

$$f_{i,j} = \frac{tf_{i,j}}{\max_l f_{l,j}} \tag{4-1}$$

df_i : Frequência de um termo t_i na coleção de documentos.

N : Número de documentos na coleção .

idf_i : Inverso da frequência do termo t_i na coleção de documentos.

$$idf_i = \log \frac{N}{df_i} \tag{4-2}$$

$tf\text{-}idf_i$: Frequência de um termo t_i e inverso da frequência do termo t_i na coleção de documentos.

$$tf\text{-}idf_i = f_{i,j} \times idf_i \tag{4-3}$$

Para entender como estes fatores são utilizados, apresenta-se um exemplo de um documento recuperado do serviço de indexação. Este documento é apresentado em C.1 no apêndice C. A estrutura do documento recuperado contém o campo **palavras** definido na tabela B.3 do apêndice B. O campo

contém um conjunto de palavras extraídas das propriedades de dado do indivíduo, durante o processamento de dados da etapa de escolhe de classes (4.3.2). Estas palavras são analisadas para calcular os fatores definidos anteriormente. A seguir apresenta-se o texto do campo **palavras**.

> **palavras:** *1996-01 Quotations in Plato's Symposium Quotations in Plato's Symposium*

O campo contém três termos significativos $termos = \{Quotations, Plato, Symposium\}$ (os números contidos no texto são descartados na análise). A frequência dos termos são $tf = \{2, 2, 2\}$ respectivamente, tendo como frequência máxima $\underset{l}{maxf} = 2$ e a frequência normalizada é $f = \{1, 1, 1\}$. Para $N = 16003$, a frequência dos termos na coleção de documentos é $df = \{6, 8, 26\}$ e o inverso da frequência dos termos $idf = \{3.4, 3.3, 2.8\}$ obtendo-se assim um *tf-idf* $= \{3.4, 3.3, 2.8\}$ para os termos, respectivamente. Dos valores *tf-idf* encontrados, o termo "Quotations" aparece como o mais relevante no documento.

Identificadas as palavras-chave do documento, é necessário responder mais duas questões:

(a) Quantas palavras-chave são encaminhadas para o índice semântico na Web?.

(b) Como as palavras-chave são combinadas para se obter um melhor resultado?

As respostas destas questões definem o volume de processamento enviado ao serviço MapReduce. Para responder a primeira questão, é importante considerar que o Sindice limita o número de resultados retornados em cada consulta, este valor é 1000. Esta restrição foi encontrada durante o desenvolvimento deste trabalho.

Cada documento correspondente ao resultado retornado pelo Sindice tem a estrutura apresentada na tabela B.4. Por outro lado, depois de uma busca é necessário considerar alguns fatores, em relação ao serviço de indexação, para persistir os documentos recuperados do Sindice: (1) o volume de cada documento retornado, (2) o número de documentos, e (3) o tempo de armazenamento dos documentos. A configuração adequada destes fatores influência no sucesso do fluxo de tarefas. Estes fatores são essenciais na arquitetura definida no capítulo 6 .

A Figura 4.8 apresenta dois tipos de abordagens para buscas de fontes relevantes, usando as palavras-chave identificadas. A seguir, uma descrição destas abordagens é apresentada:

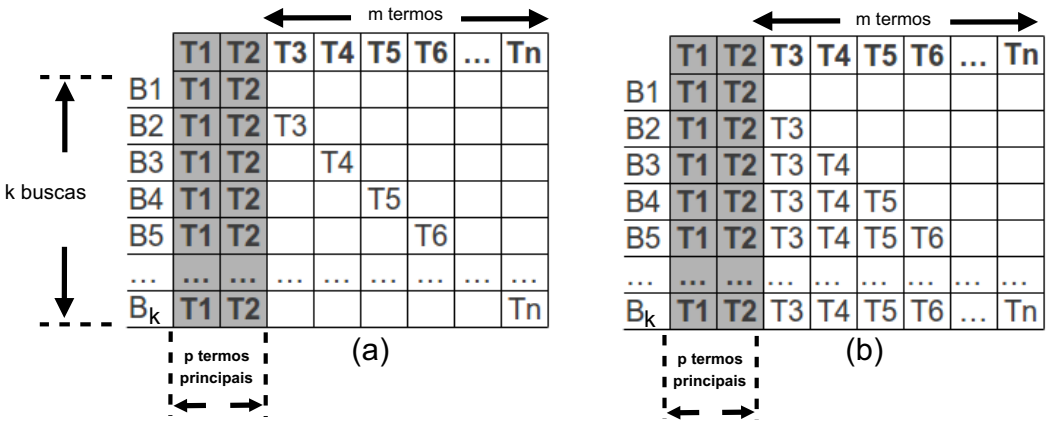

Figura 4.8: Tipos de buscas: (a) Combinação independente das palavras-chave, (b) Acúmulo de palavras-chave

(a) Combinação independente das palavras-chave: Escolhem-se as p palavras-chave com maior valor *tf-idf* do conjunto de termos extraídos do documento. Cada termo restante é combinado com as p palavras-chave identificadas, para criar uma busca independente, gerando k buscas que são enviadas ao serviço de indexação.

(b) Acúmulo de palavras-chave: Escolhem-se as p palavras-chave com maior valor *tf-idf* do conjunto de termo extraídos do documento. Cada termo restante é combinado com as p palavras-chave identificadas, para criar uma busca independente, mas cada busca mantém o termo anteriormente combinado, gerando k buscas ao serviço de indexação que são enviadas ao serviço de indexação.

A combinação das palavras chaves é feita usando o operador booleano **AND**, o que significa que cada palavra chave deve existir em todos os documentos recuperados. Isso ajuda a filtrar os resultados nas buscas executadas. Existem variáveis de configuração que definem as p palavras-chave com o maior valor de *tf-idf* e os p termos restantes para efetuar estes tipos de busca. Uma comparação destas abordagens é feita na próxima seção para definir qual delas atende melhor o processo proposto.

4.4.3 Resultados

Para os indivíduos escolhidos, das classes de "Artigo Acadêmico"e "Material Multimídia" da fonte `data.open.ac.uk`, se obtém os resultados apresentados na tabela 4.8, para as buscas definidas na figura 4.8. As buscas apresentadas foram feitas considerando as 5 palavras-chave com maior valor *tf-idf*. O

processamento das dados foi feito usando a configuração da tabela 4.7, onde um *cluster* de 5 computadores é definido.

Na figura 4.8, uma amostra de 1280 indivíduos é considerada para as buscas na coleção da indivíduos da classe Artigo Acadêmico. Para as classes de Material Multimídia é considerada uma amostra de 307 indivíduos. Os resultados revelam que a busca por combinação das palavras-chave recupera uma maior quantidade de informação. Assim, pode se afirmar que a ordem como são enviadas as palavras-chave influencia nos rankings gerados.

Tabela 4.8: Resultados da busca por combinação (BC) e por acúmulo (BA) de palavras-chave

Estatística	**Valor1 (BC)**	**Valor2 (BA)**
Artigo Acadêmico		
Amostragem de indivíduos	1280	1280
Número de fontes recuperadas	11549	9770
Número de indivíduos recuperados	59083	52142
Tempo de execução da busca	2h13min	1h55min
Material Multimídia		
Amostragem de indivíduos	307	307
Número de fontes recuperadas	14383	11489
Número de indivíduos recuperados	79459	59021
Tempo de execução da busca	1h10min	47min

4.5 Etapa de ranking

A última etapa do processo proposto é apresentado na figura 4.9, na qual o usuário pode visualizar o resultado das buscas geradas. Vale ressaltar que as buscas são geradas usando os indivíduos mais relevantes selecionados na etapa de escolha de indivíduos (na seção 4.4).

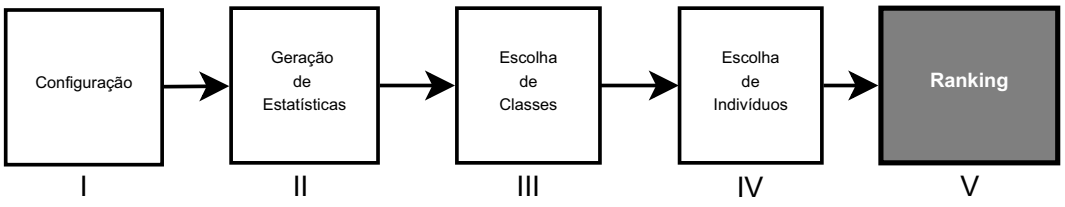

Figura 4.9: Etapa de ranking do processo proposto

As buscas definidas na figura 4.10 são executadas pelo *cluster* de 5 computadores (definido na tabela 4.7), mas o número de palavras-chave é definida em 5 (palavras-chave com maior valor *tf-idf*).

Na figura 4.10 as fontes são ordenadas pelo número de indivíduos coletados, 20 fontes são apresentadas de forma decrescente, onde a fonte com a maior quantidade de indivíduos está localizada na posição 1. Do lado esquerdo da figura encontram-se os nomes das fontes sem nenhuma ordem específica, as fontes são identificadas através de uma letra no ranking, por exemplo, a fonte DBpedia é representada pela letra **A**. Na parte superior esquerda da figura é exibida uma legenda para definir os termos usados. Esta mesma formatação é seguida nas figuras onde se apresenta um ranking de fontes.

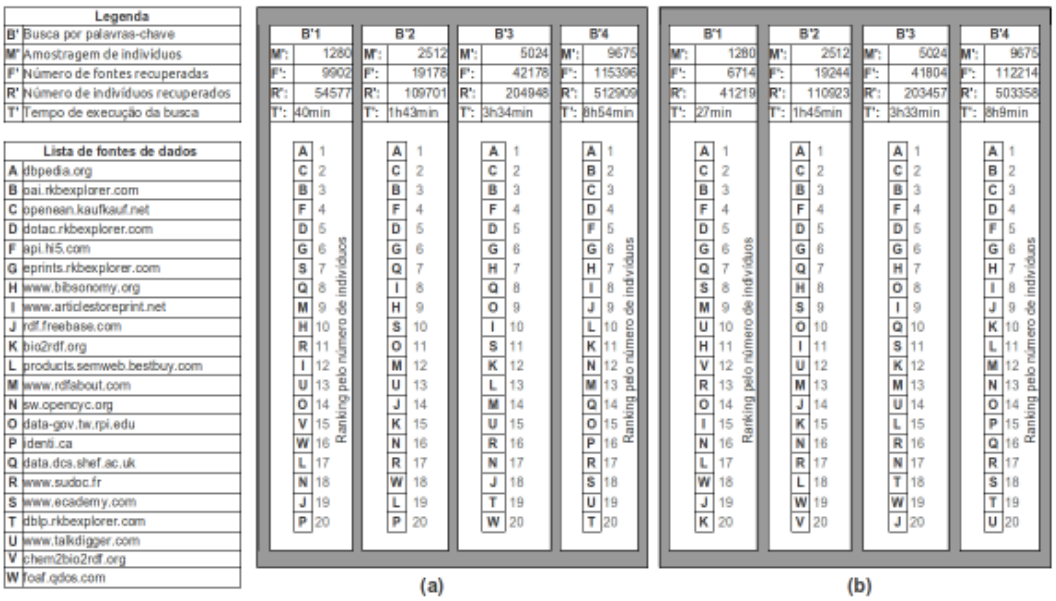

Figura 4.10: Escalabilidade do processo de busca: (a) Busca por combinação independente das palavras-chave, (b) Busca por acúmulo de palavras-chave

Na figura 4.10 observamos que o processo de busca pode escalar facilmente com diferentes amostras de indivíduos. Na figura 4.10, podemos observar que a busca por combinação independente de palavras-chave consegue extrair mais informações, se comparada com a busca por acúmulo de palavras-chave. Portanto, uma pequena variação nas palavras enviadas ao índice semântico pode retornar diferentes resultados e consequentemente afetar o processo de recomendação. Os rankings apresentado são sobre a classe Artigo Acadêmico, onde DBPedia é a fonte com maior quantidade de indivíduos.

Recursos relevantes são recuperados na busca por palavras-chave e os indivíduos da fonte `data.open.ac.uk` podem ser facilmente conectados a eles, mas é necessária uma análise especializada para identificá-los automaticamente. Neste trabalho este tipo de análise não é abrangido.

Basicamente os resultados da etapa de ranking são as saídas da etapa de escolha de indivíduos (definido na seção 4.4). Os detalhes dos resultados desta

etapa são apresentados na seção 5.3, do capítulo 5.