UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
INSTITUTO DE INFORMÁTICA
PROGRAMA DE PÓS-GRADUAÇÃO EM COMPUTAÇÃO

WILLIAN BRUNO GOMES ALVES

Formalização do Processo de Tradução de Consultas em Ambientes de Integração de Dados XML

Dissertação apresentada como requisito parcial
para a obtenção do grau de
Mestre em Ciência da Computação

Prof. Dr. Álvaro Freitas Moreira
Orientador

Porto Alegre, agosto de 2008

CIP – CATALOGAÇÃO NA PUBLICAÇÃO

Alves, Willian Bruno Gomes

Formalização do Processo de Tradução de Consultas em Ambientes de Integração de Dados XML / Willian Bruno Gomes Alves. – Porto Alegre: PPGC da UFRGS, 2008.

69 f.: il.

Dissertação (mestrado) – Universidade Federal do Rio Grande do Sul. Programa de Pós-Graduação em Computação, Porto Alegre, BR–RS, 2008. Orientador: Álvaro Freitas Moreira.

1. Integração de dados. 2. Resolução de consultas. 3. XML. I. Moreira, Álvaro Freitas. II. Título.

“The difference between a successful person and others is not a lack of strength, not a lack of knowledge, but rather a lack of will”
— VINCENT T. LOMBARDI

AGRADECIMENTOS

- Agradeço aos meus pais Ivanildo e Ana, por permitirem que eu tenha alcançado esta meta na minha vida profissional e por toda a preocupação que sempre tiveram comigo; minha irmã Karol, pelo companheirismo e apoio; e aos demais familiares, pelo carinho dado.

- Agradeço ao meu orientador, prof. Álvaro Moreira, pela disponibilidade constante durante meu curso de mestrado e pelos inúmeros e valiosos conselhos dados.

- Agradeço aos meus colegas e amigos do laboratório 202: Rodrigo, Cláudio, Rafael, Luciana, Alysson, Lincoln, Leonardo, Ramon e Germano, pelo companheirismo. Agradeço pelas sugestões dadas durante o desenvolvimento da minha dissertação e também pelo apoio e amizade.

- Agradeço ao pessoal que ingressou no mestrado junto comigo: Bruno, Jerônimo, Giovane, Raniery, Ewerton, Alex, pelos momentos de descontração, o famoso "coke time".

- Agradeço ao pessoal da Pensão do Eraldo: Kleinner, Rogério, Neila, Kênia, Candice, Douglas, Alexandre, Éddi, Weverton, Débora, Taty, Marcos Tadeu, Carlos, Flávio, Alexandre (Mineiro), Thiago, Max, Rômulo, Rafael, Cláudio, e tantos outros, por compartilharem comigo minhas angústias e minhas vitórias nessa jornada.

- Agradeço a todos que ficaram na torcida...

- Por fim, agradeço a Deus, por permitir que tudo isto fosse possível.

SUMÁRIO

LISTA DE ABREVIATURAS E SIGLAS

XPath	XML Path
BInXS	Bottom-up Integration of XML Schemata
GAV	Global-As-View
LAV	Local-As-View
GLAV	Global-And-Local-As-View
CXPath	Conceptual XPath

LISTA DE FIGURAS

LISTA DE TABELAS

RESUMO

A fim de consultar uma mesma informação em fontes XML heterogêneas seria desejável poder formular uma única consulta em relação a um esquema global conceitual e então traduzi-la automaticamente para consultas XML para cada uma das fontes. *CXPath (Conceptual XPath)* é uma proposta de linguagem para consultar fontes XML em um nível conceitual. Essa linguagem foi desenvolvida para simplificar o processo de tradução de consultas em nível conceitual para consultas em nível XML. Ao mesmo tempo, a linguagem tem como objetivo a facilidade de aprendizado de sua sintaxe. Por essa razão, sua sintaxe é bastante semelhante à da linguagem *XPath* utilizada para consultar documentos XML.

Nesta dissertação é definido formalmente o mecanismo de tradução de consultas em nível conceitual, escritas em *CXPath*, para consultas em nível XML, escritas em *XPath*. É mostrado o tratamento do relacionamento de herança no mecanismo de tradução, e é feita uma discussão sobre a relação entre a expressividade do modelo conceitual e o mecanismo de tradução.

Existem situações em que a simples tradução de uma consulta *CXPath* não contempla alguns resultados, pois as fontes de dados podem ser incompletas. Neste trabalho, o modelo conceitual que constitui o esquema global do sistema de integração de dados é estendido com dependências de inclusão e o mecanismo de resolução de consultas é modificado para lidar com esse tipo de dependência. Mais especificamente, são apresentados mecanismos de reescrita e eliminação de redundâncias de consultas a fim de lidar com essas dependências.

Com o aumento de expressividade do esquema global é possível inferir resultados, a partir dos dados disponíveis no sistema de integração, que antes não seriam contemplados com a simples tradução de uma consulta.

Também é apresentada a abordagem para integração de dados utilizada nesta dissertação de acordo com o arcabouço formal para integração de dados proposto por (LENZERINI, 2002). De acordo com o autor, tal arcabouço é geral o bastante para capturar todas as abordagens para integração de dados da literatura, o que inclui a abordagem aqui mostrada.

Palavras-chave: Integração de dados, resolução de consultas, XML.

Formalization of a Query Translation Process in XML Data Integration

ABSTRACT

In order to search for the same information in heterogeneous XML data sources, it would be desirable to state a single query against a global conceptual schema and then translate it automatically into an XML query for each specific data source. *CXPath* (for *Conceptual XPath*) has been proposed as a language for querying XML sources at the conceptual level. This language was developed to simplify the translation process of queries at conceptual level to queries at XML level. At the same time, one of the goals of the language design is to facilitate the learning of its syntax. For this reason its syntax is similar to the *XPath* language used for querying XML documents.

In this dissertation, a translation mechanism of queries at conceptual level, written in *CXPath*, to queries at XML level, written in *XPath*, is formally defined. The inheritance relationship in the translation mechanism is shown, being discussed the relation between the conceptual model expressivity and the translation mechanism.

In some cases, the translation of a *CXPath* query does not return some of the answers because the data sources may be incomplete. In this work, the conceptual model, which is the basis for the data integration system's global schema, is improved with inclusion dependencies, and the query answering mechanism is modified to deal with this kind of dependency. More specifically, mechanisms of query rewriting and redundancy elimination are presented to deal with this kind of dependency.

This global schema improvement allows infer results, with the data available in the system, that would not be provided with a simple query translation.

The approach of data integration used in this dissertation is also presented within the formal framework for data integration proposed by (LENZERINI, 2002). According to the author, that framework is general enough to capture all approaches in the literature, including, in particular, the approach considered in this dissertation.

Keywords: Data integration, query answering, XML.

1 INTRODUÇÃO

Integração de dados consiste em prover um acesso uniforme a um conjunto de fontes de dados distribuídas, heterogêneas e autônomas. Os sistemas de integração de dados de interesse neste trabalho são caracterizados por uma arquitetura baseada em um esquema global e um conjunto de fontes. As fontes possuem os dados de fato, e o esquema global provê uma visão integrada e virtual das fontes envolvidas (LENZERINI, 2002).

Os sistemas de integração de dados isentam o usuário do conhecimento de quais fontes possuem os dados de interesse, como os dados estão estruturados nas fontes, e como eles são combinados para responder às consultas dos usuários, formuladas em termos do esquema global.

Integrar fontes de dados heterogêneas é um problema fundamental em banco de dados que tem sido estudado amplamente na últimas décadas, tanto do ponto de vista formal quanto do ponto de vista prático (HAAS, 2007). Este importante problema emerge em uma variedade de situações, tanto em aplicações comerciais (por exemplo, quando duas ou mais companhias precisam combinar seus dados), quanto científicas (por exemplo, combinar resultados de pesquisas de diferentes repositórios de bio-informática).

Usuários submetem consultas ao esquema global imaginando a existência de uma base global para tal esquema. Porém, as fontes são acessadas através de mapeamentos, a fim de responder as consultas dos usuários. A Figura 1.1 apresenta os principais componentes da abordagem para integração de dados utilizada nesta dissertação: o esquema global, as fontes, e os mapeamentos.

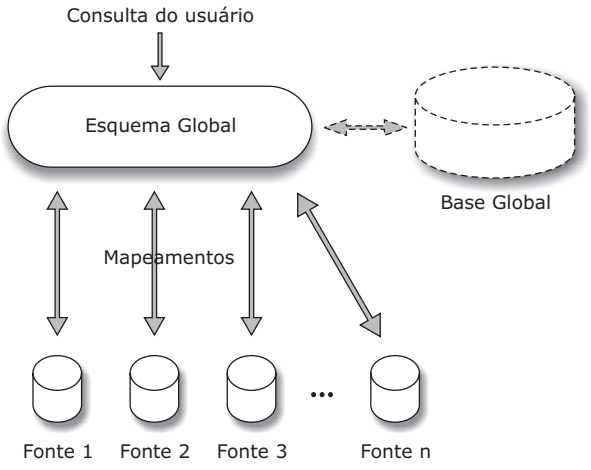


Figura 1.1: Integração de dados baseada em esquema global.

Neste trabalho é utilizada uma abordagem para integração de dados, desenvolvida no Instituto de Informática da UFRGS, que utiliza um modelo conceitual como esquema global, criado a partir da integração semântica dos esquemas XML das fontes (MELLO; HEUSER, 2005). Nessa abordagem é adotado um modelo conceitual para definição do esquema global ao invés do modelo de dados XML, porque o modelo XML é incapaz de abstrair vários esquemas XML ao mesmo tempo. Isso se dá pela natureza hierárquica dos dados XML. Fontes XML heterogêneas pertencentes ao mesmo domínio de aplicação podem ter diferentes representações hierárquicas do mesmo relacionamento *muitos-para-muitos*. Um modelo conceitual representa diretamente os relacionamentos *muitos-para-muitos* sem impor uma ordem de navegação.

Um ponto crucial na integração de dados é estabelecer a especificação da correspondência entre as fontes e o esquema global (mapeamentos). É exatamente essa correspondência que irá determinar como as consultas feitas ao sistema serão resolvidas.

O trabalho (MELLO; HEUSER, 2005) segue a abordagem GAV (*global-as-view*) para especificação dos mapeamentos. Por essa abordagem o esquema global representa uma visão das fontes de dados. Para cada elemento do esquema global é associada uma visão sobre as fontes, especificando explicitamente como extrair os dados das fontes, o que favorece a resolução de consultas. Especificamente no trabalho citado, para cada elemento do modelo conceitual são especificadas consultas *XPath* relacionando os elementos do modelo conceitual com elementos das fontes.

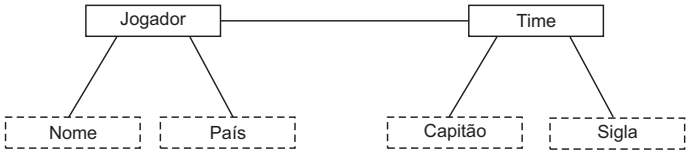
Em (CAMILLO; HEUSER; MELLO, 2003) é proposta a linguagem de consulta *CXPath (Conceptual XPath)*, que atua sobre modelos conceituais. Naquele trabalho também é mostrado, através de exemplos, o processo de tradução de consultas globais *(CXPath)* para consultas locais *(XPath)*, considerando cada fonte de forma independente, ou seja, não é tratada a tradução de consultas que precisam ser decompostas sobre várias fontes a fim de buscar a informação desejada.

Em (FEIJÓ et al., 2007) a linguagem *CXPath* é estendida com a utilização de herança e auto-relacionamento. Nesse trabalho é especificado formalmente um critério para validação de consultas *CXPath* de acordo com um modelo conceitual. Por fim, em (SILVEIRA; HEUSER, 2007) é tratado o problema de decompor uma consulta global em consultas locais, permitindo a interação entre várias fontes para construção da resposta.

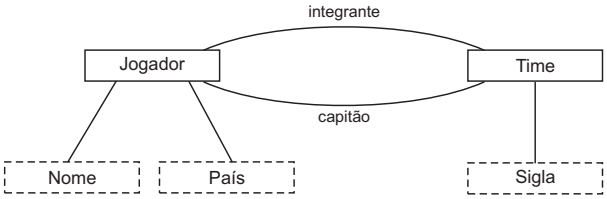
As contribuições deste trabalho são as seguintes:

(i) Formalização do mecanismo de tradução de consultas em nível conceitual, expressas em *CXPath*, para consultas em nível XML, escritas em *XPath*. Em (CAMILLO; HEUSER; MELLO, 2003) esse processo de tradução é mostrado através de exemplos, porém ele não é definido formalmente. Neste trabalho é mostrado um conjunto de regras que, utilizando informações de mapeamento, traduzem uma consulta *CXPath* em consultas *XPath*;

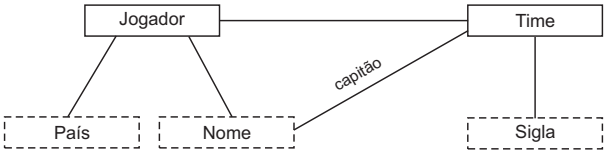
(ii) O poder de expressão do modelo conceitual é aumentado com a possibilidade de especificação de dependências de inclusão sobre o modelo conceitual, sendo definido um processo de reescrita de consultas *CXPath* que leva em consideração essas dependências. O esquema global é a representação do domínio de interesse do sistema de integração de dados: restrições de integridade são expressas nesse esquema com o intuito de aumentar sua expressividade, aumentando assim sua capacidade de representar o mundo real. Com esse aumento de expressividade é possível inferir resultados, a partir dos dados disponíveis nas fontes, que antes não seriam contemplados com a simples tradução de uma consulta;



(a) Modelo conceitual com redundância



(b) Modelo conceitual sem redundância 1



(c) Modelo conceitual sem redundância 2

Figura 1.2: Modelos conceituais com e sem redundância.

(iii) Por fim, a abordagem para integração de dados utilizada neste trabalho é apresentada utilizando o arcabouço formal proposto por (LENZERINI, 2002). Este arcabouço é bastante difundido na área de integração de dados e utilizado em diversos trabalhos (POGGI; ABITEBOUL, 2005; CALÌ et al., 2003; CALÌ et al., 2004; CALVANESE et al., 2001; LOPATENKO, 2004; EITER, 2005; LEONE et al., 2005). De acordo com o próprio autor, tal arcabouço é geral o bastante para capturar todas as abordagens de integração encontradas na literatura, o que inclui a abordagem para integração de dados utilizada nesta dissertação.

É importante salientar que as dependências de inclusão utilizadas neste trabalho servem para sanar um problema de concepção do modelo conceitual. Como o BInXS é um processo semi-automático, a construção do modelo depende do nível do usuário que interage com o processo. Por exemplo, a Figura 1.2 mostra modelos conceituais sobre o mesmo domínio. Os modelos 1.2(b) e 1.2(c) estão conceitualmente corretos enquanto que o modelo 1.2(a) possui redundância, pois um capitão de time representa um nome de jogador. É nesse tipo de caso que as dependências de inclusão são uma contribuição neste trabalho.

Assim como em (CAMILLO; HEUSER; MELLO, 2003), cada fonte é tratada de forma independente, sendo assim, neste trabalho o processo de tradução não leva em consideração dados fragmentados, ou seja, não é abordada a tradução de consultas que precisam ser decompostas sobre várias fontes afim de buscar a informação desejada.

Outra característica significativa neste trabalho é que as fontes de dados são consideradas incompletas, ou seja, elas não provêem toda a informação necessária para responder às consultas dos usuários.

Este trabalho está estruturado da seguinte forma: o Capítulo 2 apresenta as informações necessárias para compreensão deste trabalho, mostrando os componentes utilizados na abordagem para integração de dados adotada nesta dissertação, o arcabouço formal para integração de dados, e a abordagem para integração apresentada, inserida no arcabouço formal para integração de dados. No Capítulo 3 é apresentado o mecanismo de tradução de consultas *CXPath* para consultas *XPath*; no Capítulo 4 são apresentadas as dependências de inclusão e o mecanismo de reescrita de consultas *CXPath* em outras consultas *CXPath* utilizando as informações dessas dependências. No Capítulo 5 são apresentadas as conclusões do trabalho e uma discussão sobre trabalhos futuros. Por fim, o Apêndice A apresenta noções básicas sobre XML e *XPath*. No restante desta introdução

serão apresentados os trabalhos relacionados a esta dissertação.

Trabalhos Relacionados

O sistema de integração de dados XML e relacionais Agora (MANOLESCU et al., 2001) utiliza um esquema XML como esquema global, expresso através de uma DTD e sem a possibilidade de especificação de restrições de integridade. Consultas submetidas ao esquema global são expressas utilizando a linguagem de consulta *XQuery*. Esse sistema é caracterizado por um conjunto de mapeamentos seguindo a abordagem LAV (*local-as-view*), ou seja, cada fonte é considerada uma visão do esquema global. Além disso, as fontes são consideradas incompletas. Os mapeamentos são definidos em termos de um esquema intermediário relacional, virtual e genérico, que modela a estrutura genérica de um esquema global XML, ao invés de em termos do próprio esquema global XML. O processamento de consultas nesse sistema é baseado na reescrita de consultas que é realizada primeiramente através da tradução da consulta para o esquema relacional genérico e depois utilizando técnicas tradicionais para resolução de consultas utilizando visões.

Em (CALÌ et al., 2001) é apresentado um sistema de integração de dados que adota um modelo conceitual como esquema global. Os mapeamentos nesse trabalho seguem a abordagem GAV e cada fonte de dados relacional é considerada incompleta. O modelo conceitual desse trabalho utiliza as características básicas do modelo ER, estendido com a possibilidade de especificação de herança tanto sobre as entidades quanto sobre os relacionamentos do esquema. Nesse trabalho é mostrado que a resolução de consultas, mesmo utilizando a abordagem GAV, está intimamente relacionada com a resolução de consultas em bancos de dados incompletos. Essa situação ocorre por causa da existência de condições semânticas (restrições) impostas pelo modelo conceitual especificado nesse trabalho. Nesse trabalho é apresentado um algoritmo para processar consultas capaz de adicionar resultados àqueles extraídos das fontes, explorando as condições semânticas expressas sobre o modelo conceitual global. Para tal é mostrado um conjunto de regras para que as condições semânticas sejam capturadas do modelo conceitual e incorporadas às consultas globais.

DIS@DIS (CALÌ et al., 2004) é um sistema de integração semântica de fontes de dados relacionais acessado através de um esquema global também relacional. Esse sistema permite a especificação tanto de mapeamentos LAV quanto mapeamentos GAV. DIS@DIS assume que as fontes são incompletas, ou seja, elas podem não prover toda a informação necessária para responder consultas submetidas ao sistema de integração. Esse sistema permite a especificação de restrições de integridade (dependências de chave, inclusão e exclusão) sobre o esquema global. As restrições de integridade, assim como nesta dissertação, são levadas em consideração na resolução de consultas, reformulando cada consulta do usuário em uma nova consulta na qual são incluídas as informações das restrições.

IBIS (*Internet-Based Information System*) (CALì et al., 2003) é um sistema para integração semântica de fontes heterogêneas, capaz de resolver consultas sob restrições de integridade. IBIS utiliza um esquema global relacional para consultar dados nas fontes e é capaz de lidar com uma variedade de fontes de dados heterogêneas incluindo fontes provenientes da Web, relacionais e sistemas legados. Cada fonte não relacional é adaptada a fim de prover uma visão relacional dela mesma. IBIS segue a abordagem para especificação dos mapeamentos GAV e cada fonte é considerada incompleta. O sistema permite a especificação de restrições de integridade no esquema global e considera a presença de

restrições sobre as fontes com o intuito de otimizar a extração dos dados. Em particular, chaves primárias e estrangeiras podem ser especificadas sobre o esquema global, e dependências funcionais e dependências de inclusão *full-width* (inclusão total entre relações) podem ser especificadas sobre as fontes.

Em (POGGI; ABITEBOUL, 2005) é proposta uma abordagem para integração de dados baseada em um esquema global XML e um conjunto de fontes de dados XML. O esquema global é caracterizado por um conjunto de restrições, expressas através de uma DTD e de restrições de integridade XML, como definidas em (FAN et al., 2001). Os mapeamentos são definidos utilizando a abordagem LAV e são especificados através de uma extensão das *ps-queries* (ABITEBOUL; SEGOUFIN; VIANU, 2006), uma linguagem de consultas simples em árvores. Nesse trabalho também é definida uma função que identifica unicamente nós provenientes de diferentes fontes XML. São propostos também algoritmos de resolução de consultas sobre diferentes suposições para os mapeamentos sendo realizado um estudo de suas complexidades.

O trabalho (AMANN et al., 2002) descreve uma abordagem para integração de dados XML, baseada em uma ontologia como interface de acesso as fontes XML, e seguindo a abordagem LAV para especificação dos mapeamentos. Nesse trabalho é apresentado uma linguagem simples para definir documentos XML como visões do esquema global, bem como uma abordagem para o processamento de consultas que inclui a reescrita de consultas, baseada nos mapeamentos, dos termos do esquema global em uma ou mais consultas XML sobre as fontes locais. Similar ao trabalho (SILVEIRA; HEUSER, 2007) esse trabalho utiliza o conceito de chaves para identificar unicamente objetos possibilitando assim a decomposição de consultas. A linguagem de consultas é baseada nas cláusulas *select-from-where* seguindo uma sintaxe similar a OQL, diferente da linguagem *CXPath* que é baseada na navegação entre conceitos em um grafo, similar a linguagem de consultas em árvores *XPath*.

A reescrita de consultas possui bastante relevância em uma variedade de problemas em bancos de dados: otimização de consultas (XU; ÖZSOYOGLU, 2005; DEUTSCH; POPA; TANNEN, 2006), independência física dos dados (DEUTSCH; POPA; TANNEN, 1999), controle de acesso a documentos (FAN et al., 2004), integração de dados (CALÌ et al., 2003; AMANN et al., 2002; YU; POPA, 2004), entre outros. Uma visão geral e aplicações do problema de reescrever consultas utilizando visões pode ser visto em (HALEVY, 2001).

Em (CALÌ et al., 2003) é abordado a reescrita de consultas quando dependências de chave e inclusão são especificadas sob o esquema global. Nesse trabalho é considerada a abordagem GAV para especificação dos mapeamentos e as fontes são consideras incompletas. É apresentado um algoritmo para reescrever consultas submetidas ao esquema global de forma a tratar as restrições nele especificadas.

Esse trabalho se assemelha bastante com a abordagem para resolução de consultas utilizando restrições de integridade proposta nesta dissertação. Porém, em (CALÌ et al., 2003) é utilizado o modelo relacional no esquema global e nas fontes, ao invés de um modelo conceitual e fontes XML. Outra diferença é que a reescrita de consultas desse trabalho é realizada sobre a linguagem de consulta *union of conjunctive queries (UCQ)* (ABITEBOUL; HULL; VIANU, 1995), e neste trabalho é utilizada a linguagem *CXPath*.

Assim como a abordagem proposta nessa dissertação, em (CALÌ et al., 2003) cada consulta primeiramente é reescrita de forma a incorporar a informação das restrições na consulta global e somente depois ela é traduzida para fontes.

Neste trabalho foi necessário a definição das dependências de inclusão sobre o modelo conceitual, o que em (CALÌ et al., 2003) não foi necessário se aproveitando das definições clássicas propostas para o modelo relacional (ABITEBOUL; HULL; VIANU, 1995).

O trabalho (YU; POPA, 2004) propõe um algoritmo de resolução de consultas XML sob um sistema de integração de dados relacionais e XML, no qual o esquema global é caracterizado por um conjunto de restrições expressivas, chamadas *nested equality-generating dependencies (NEGDs)*, que inclui dependências funcionais em esquemas relacionais ou aninhados, restrições de chave do XML Schema e restrições mais gerais declarando que certas tuplas/elementos do esquema global devem satisfazer certas igualdades. Nesse trabalho essas restrições podem ser utilizadas como regras para combinar dados de múltiplas fontes com informação sobreposta. Os mapeamentos são especificados por uma linguagem utilizada na ferramenta Clio (POPA et al., 2002), seguindo a abordagem GLAV (*global-and-local-as-view*), ou seja, possibilita a especificação tanto de mapeamentos GAV quanto LAV, e as fontes são consideras incompletas. Esse trabalho propõe dois algoritmos: o *basic query rewrite* que reformula as consultas submetidas ao esquema global em termos das fontes, baseado nas informações de mapeamentos, enquanto que o *query resolution* gera reescritas adicionais de forma a levar em consideração as restrições especificadas no esquema global.

2 DEFINIÇÕES PRELIMINARES

Este capítulo tem como objetivo introduzir os conceitos utilizados ao longo desta dissertação. Primeiramente, nas Seções 2.1-2.4 são mostrados os principais componentes da abordagem para integração de dados utilizada neste trabalho. Na Seção 2.5 é mostrado o arcabouço formal para integração de dados proposto por (LENZERINI, 2002), e por fim, na Seção 2.6 é apresentada a abordagem para integração de dados adotada neste trabalho no arcabouço formal mostrado.

2.1 Esquema Global

O esquema global utilizado nesta abordagem para integração de dados é representado através de um modelo conceitual. Esse modelo é construído através de uma abordagem *bottom-up* e semi-automática para a integração semântica de esquemas XML, chamada BInXS (MELLO, 2002; MELLO; HEUSER, 2005).

O processo de integração é semi-automático, pois requer a intervenção de um especialista para ajustar a semântica dos dados no processo de integração. É uma abordagem *bottom-up*, pois constrói um esquema global a partir de um conjunto de esquemas XML heterogêneos, sob o mesmo domínio de aplicação.

O modelo conceitual gerado é uma versão simplificada do modelo ORM (HALPHIN, 1998). Esse modelo é baseado em dois tipos de *conceitos*: *léxicos* e *não-léxicos*. Conceitos léxicos representam objetos que possuem um conteúdo textual. Esse tipo de conceito abstrai elementos atômicos XML. Conceitos não-léxicos não possuem um representação textual e são abstrações de elementos XML que contém outros elementos.

O modelo conceitual permite relacionamentos de associação (com a definição de papéis) com restrições de cardinalidade, e relacionamentos de herança. Na Figura 2.1 é apresentado um exemplo de modelo conceitual, com informações de cardinalidade e relacionamentos de associação sem nome, construído a partir dos esquemas XML da Figura 2.2. Os conceitos léxicos são representados através de retângulos tracejados e os conceitos não-léxicos através de retângulos contínuos.

É adotado um modelo conceitual para definição do esquema global ao invés do modelo XML, porque o modelo XML é incapaz de abstrair vários esquema XML ao mesmo tempo. Isso se dá pela natureza hierárquica dos dados XML. Fontes XML heterogêneas pertencentes ao mesmo domínio de aplicação podem ter diferentes representações hierárquicas do mesmo relacionamento *muitos-para-muitos*. Um modelo conceitual representa diretamente os relacionamentos *muitos-para-muitos* sem impor uma ordem de navegação.

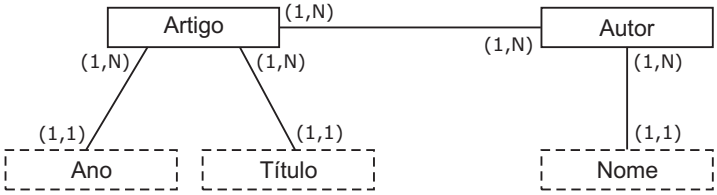
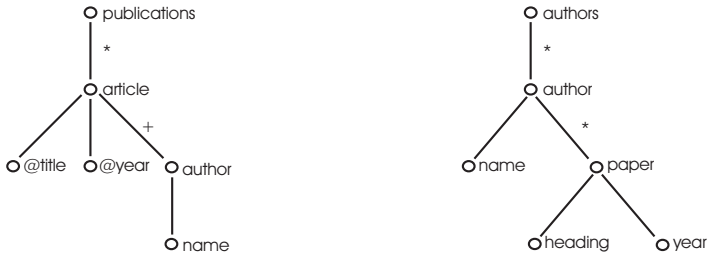


Figura 2.1: Exemplo de modelo conceitual.



(a) Esquema XML 1 (b) Esquema XML 2

Figura 2.2: Exemplos de esquemas XML.

2.2 Base Global

Uma *base global* para um sistema de integração é uma base de dados abstrata que contém instâncias de um dado esquema global, respeitando portanto as restrições desse esquema. Os usuários submetem consultas ao esquema global imaginando a existência de um banco de dados para tal esquema mas, na verdade, o sistema de integração se encarrega de acessar as fontes relevantes e retornar os resultados de uma consulta.

Nesta dissertação, em uma base global as instâncias de conceitos não-léxicos são representadas através de um identificador e pelo nome do conceito ao qual pertence aquela instância. Já as instâncias de conceitos léxicos são representadas pelo nome do conceito associado juntamente com seu valor. Um par, cujo os elementos são instâncias de conceitos léxicos e/ou conceitos não-léxicos, representam uma instância de um relacionamento. Uma definição formal para bases de dados globais é apresentada em (SILVEIRA, 2006).

Uma *base global válida* para um determinado sistema de integração de dados é uma base global para o esquema global do sistema e que satisfaz os mapeamentos. Os conceitos *base global* e *base global válida* para um sistema de integração serão definidos com precisão na Subseção 2.5.

A Figura 2.3 mostra uma representação gráfica de uma base global para o modelo conceitual apresentado na Figura 2.1 referente às fontes XML da Figura 2.4. Os círculos representam instâncias de conceitos, enquanto as linhas representam instâncias de relacionamentos. Na verdade, para dizer que a base da Figura 2.3 é uma base global válida (para o sistema de integração constituído pelo esquema global da Figura 2.1, os esquemas das fontes da Figura 2.2 e pelas fontes da Figura 2.4) é necessário também levar em conta os mapeamentos. Em outras palavras, os mapeamentos devem "permitir" a base da Figura 2.3. Os mapeamentos serão discutidos precisamente na Subseção 2.4.

As definições de base global e de base global válida são de suma importância para definição da semântica de sistemas de integração de dados e para definição da semântica da resolução de consultas desses sistemas.

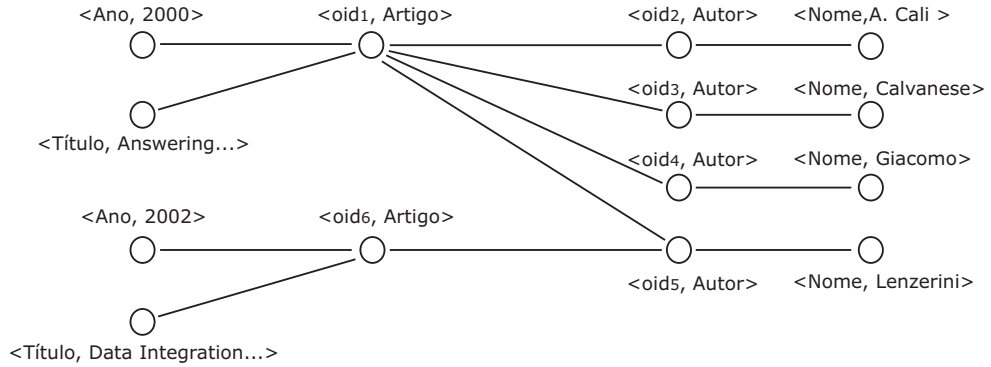
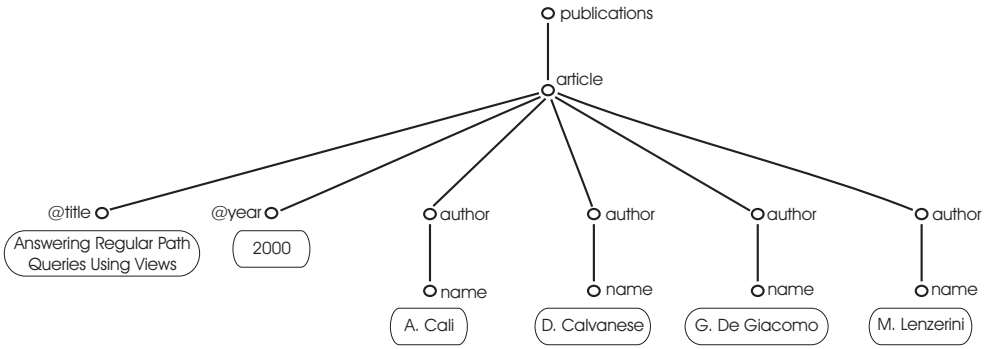
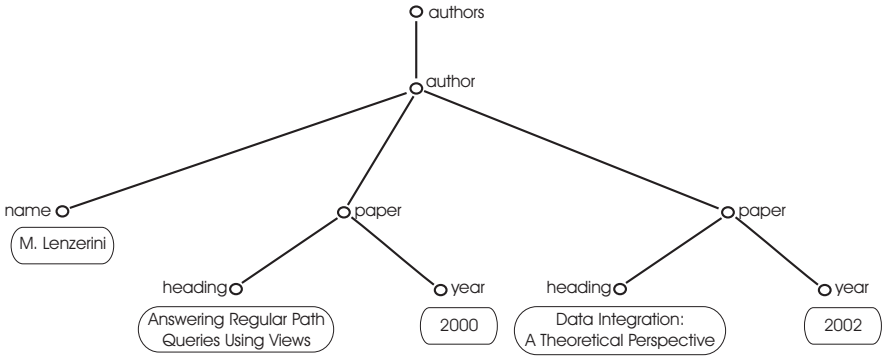


Figura 2.3: Exemplo de base global para o modelo conceitual da Figura 2.1.



(a) Fonte XML referente ao esquema XML da Figura 2.2(a)



(b) Fonte XML referente ao esquema XML da Figura 2.2(b)

Figura 2.4: Exemplos de fontes XML para os esquemas da Figura 2.2

2.3 Linguagem de Consulta

A linguagem de consulta utilizada nesta abordagem para integração de dados é a linguagem *CXPath (Conceptual XPath)* (CAMILLO; HEUSER; MELLO, 2003). *CXPath* foi desenvolvida para realizar consultas no nível conceitual com dois objetivos principais: *i)* simplificar o processo de tradução de uma consulta no nível conceitual (*esquema global*) para consultas no nível XML (*fontes*); e *ii)* simplificar o processo de aprendizagem da linguagem para aqueles familiarizados com linguagens de consulta do padrão XML, como por exemplo *XPath* (XML PATH LANGUAGE (XPath) VERSION 1.0 - W3C RECOMENDATION 16 NOVEMBER 1999, 1999) e *XQuery* (XQUERY 1.0: AN XML QUERY LANGUAGE - W3C RECOMENDATION 23 JANUARY 2007, 2007).

Apesar de serem sintaticamente semelhantes, *CXPath* e *XPath* possuem semânticas diferentes, pois são aplicadas sobre diferentes modelos de dados. Enquanto que uma expressão de caminho em *XPath* especifica a navegação através de relacionamentos hierárquicos, em *CXPath*, uma expressão de caminho especifica a navegação através de uma rede de relacionamentos entre conceitos.

Em (FEIJÓ et al., 2007) foi proposta a especificação formal de um critério para validação de consultas *CXPath* contra um modelo conceitual. A formalização resultante fornece uma descrição concisa e precisa da linguagem, que pode ser utilizada por desenvolvedores e elaboradores de consultas globais.

A gramática da linguagem *CXPath* é apresentada na Figura 2.5. Pela gramática apresentada pode-se notar a semelhança com a linguagem *XPath* abreviada. Consultas *CXPath* representam a navegação entre conceitos de um modelo conceitual, podendo ser expressões de caminho absolutas ($AbsolutePath$) ou relativas ($RelativePath$). Assim como em *XPath*, predicados podem ser definidos nas consultas, especificados entre colchetes [ ].

A principal diferença está na possibilidade de qualificação da navegação entre os conceitos em *CXPath*. Pela gramática, os relacionamentos podem ser representados através de um nome $RelationshipName$ (juntamente com um papel desempenhado por um conceito naquele relacionamento $RelationshipName.RoleNome$), ou a navegação pode ser feita através de um relacionamento não identificado, representado por ε.

Abaixo, são apresentados alguns exemplos de consultas *CXPath*, levando em consideração o modelo conceitual da Figura 2.1 com o intuito de demonstrar a linguagem.

Exemplo 2.1 *Retornar o nome dos autores de artigos.*

```
/Artigo/Autor/Nome
```

No Exemplo 2.1 ocorre uma navegação simples entre conceitos de um modelo conceitual. A consulta retornará todos os elementos do conceito `Nome` relacionados ao conceito `Autor`, que por sua vez está relacionado ao conceito `Artigo`. A navegação dos conceitos é realizada da esquerda para direita, nesse caso, do conceito `Artigo` para o conceito `Autor` e depois para o conceito `Nome` através de relacionamentos não identificados.

Exemplo 2.2 *Retornar os artigos publicados no ano 2000.*

```
/Artigo[Ano=2000]
```

CXPath também permite a especificação de condições que tornam possível a filtragem dos dados. Assim como em *XPath*, os predicados em *CXPath* determinam uma condição

$$
\begin{array}{lcl}
CXPath & ::= & AbsolutePath \\
 & | & RelativePath \\
 & & \\
AbsolutePath & ::= & / \\
 & | & / \ RelativePath \\
 & & \\
RelativePath & ::= & Relationship \ id \ Predicates \\
 & | & Relationship \ id \ Predicates \ / \ RelativePath \\
 & & \\
Relationship & ::= & \{ \ RelationshipName \ \} \\
 & | & \{ \ RelationshipName.RoleName \ \} \\
 & | & \varepsilon \\
 & & \\
Predicates & ::= & [/RelativePath_1 \ op \ /RelativePath_2] \ Predicates \\
 & | & [/RelativePath_1 \ op \ RelativePath_2] \ Predicates \\
 & | & [/RelativePath \ op \ Literal] \ Predicates \\
 & | & [\ RelativePath_1 \ op \ /RelativePath_2] \ Predicates \\
 & | & [\ RelativePath_1 \ op \ RelativePath_2] \ Predicates \\
 & | & [\ RelativePath \ op \ Literal] \ Predicates \\
 & | & \varepsilon \\
 & & \\
Literal & ::= & IntegerLiteral \\
 & | & StringLiteral \\
 & & \\
op & ::= & = \ | \ ! = \ | \ < \ | \ <= \ | \ > \ | \ >=
\end{array}
$$

Figura 2.5: Gramática de CXPath.

lógica que deve ser atendida. No Exemplo 2.2, somente serão retornados os elementos do conceito `Artigo` cujos elementos do conceito `Ano`, relacionado a `Artigo`, possuam o valor '2000'.

Exemplo 2.3 *Retornar o nome dos primeiros autores de artigos.*

```
/Artigo/{primeiro_autor}Autor/Nome
```

No Exemplo 2.3, são selecionados os elementos do conceito `Nome` relacionados com o conceito `Autor`, que por sua vez está relacionado com o conceito `Artigo`. A navegação do conceito `Artigo` para o conceito `Autor` deve ser realizada através do relacionamento qualificado `primeiro_autor`, restringindo o resultado da consulta a apenas nomes de primeiros autores de artigos.

Logicamente, as consultas acima especificadas podem não estar de acordo com o modelo conceitual da Figura 2.1. No Exemplo 2.3, por exemplo, a consulta está sintaticamente correta, porém não está de acordo com o modelo conceitual apresentado. Não existe um relacionamento de nome `primeiro_autor` entre os conceitos `Autor` e `Artigo`. Porém, se esta consulta for submetida ao modelo conceitual da Figura 2.7 ela estará de acordo com o modelo. Como já salientado nesta dissertação, a verificação de consultas *CXPath* contra um modelo conceitual é realizada em (FEIJÓ et al., 2007).

2.4 Fontes de Dados e Mapeamentos

O BInXS (MELLO; HEUSER, 2005) utiliza duas fases para realizar a construção de um modelo conceitual a partir dos *esquemas XML das fontes*: conversão de esquemas e unificação. A primeira fase mapeia cada esquema XML envolvido no processo de integração em um modelo conceitual correspondente. A segunda fase, unifica os modelos conceituais gerados na etapa anterior, construindo de fato o modelo conceitual global utilizado como interface de acesso às fontes.

As informações de mapeamento são definidas durante a fase de *conversão de esquemas* para cada conceito e relacionamento do modelo conceitual. BInXS suporta uma estratégia de mapeamentos baseado em expressões *XPath* de forma a manter as correspondências entre elementos do modelo conceitual e os elementos dos esquemas XML das fontes.

O mapeamento de conceitos do modelo conceitual é definido através de expressões de caminho absolutas em *XPath*, ou seja, um caminho desde o elemento *raíz* até o elemento ou atributo correspondente em um esquema XML. Na Tabela 2.1 são apresentadas as informações de mapeamento dos conceitos do modelo conceitual da Figura 2.1 para o esquema XML da Figura 2.2(a).

Tabela 2.1: Mapeamento de conceitos para o esquema XML da Figura 2.2(a).

Conceitos	Fonte XML
Artigo	`/publications/article`
Autor	`/publications/article/author`
Nome	`/publications/article/author/name`
Título	`/publications/article/@title`
Ano	`/publications/article/@year`

O mapeamento de relacionamentos é definido através de expressões de caminho relativas em *XPath*. Tais expressões explicitam como navegar entre elementos em um esquema XML. Como o modelo conceitual é um grafo é possível navegar entre conceitos em qualquer direção, por esse motivo, os mapeamentos são definidos para ambas as direções de um relacionamento (por exemplo, na Tabela 2.2 existe uma informação de mapeamento Artigo ε Autor e também a informação Autor ε Artigo). Na Tabela 2.2 são apresentadas as informações de mapeamento dos relacionamentos do mesmo modelo conceitual utilizado anteriormente e para mesma fonte. Note que ε significa relacionamento sem nome.

Quando não é possível relacionar um elemento do modelo conceitual com um elemento do esquema XML, a informação de mapeamento é especificada como `indefinido`. Por exemplo, considerando o esquema XML da Figura 2.6, com informação apenas sobre artigos, e o modelo conceitual da Figura 2.1, a Tabela 2.3 apresenta as informações de mapeamento dos conceitos.

Observe que não há como mapear os conceitos Autor e Nome na fonte da Figura 2.6.

A Tabela 2.4 mostra as informações de mapeamento sobre os conceitos envolvidos nos relacionamentos identificados do modelo conceitual da Figura 2.7 construído a partir da esquema XML da Figura 2.8.

Cada informação de mapeamento é especificada de acordo com sua suposição. Essa suposição indica qual a precisão de cada mapeamento, relacionando as instâncias das fontes com as instâncias do esquema global. Sendo assim, se a consulta *XPath* associada a

Tabela 2.2: Mapeamento de relacionamentos para o esquema XML da Figura 2.2(a).

Relacionamentos	Fonte XML
Artigo ε Autor	`author`
Autor ε Artigo	..
Artigo ε Título	`@title`
Título ε Artigo	..
Artigo ε Ano	`@year`
Ano ε Artigo	..
Autor ε Nome	`nome`
Nome ε Autor	..

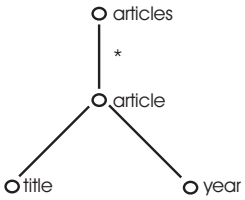


Figura 2.6: Esquema XML com informação sobre artigos.

Tabela 2.3: Mapeamento de conceitos para o esquema XML da Figura 2.6.

Conceitos	Fonte XML
Artigo	`/articles/article`
Título	`/articles/article/title`
Ano	`/articles/article/year`
Autor	`indefinido`
Nome	`indefinido`

Tabela 2.4: Informações de mapeamento para o esquema XML da Figura 2.8.

Conceitos/Relacionamentos	Fonte XML
Artigo	`/artigos/artigo`
Autor	`/artigos/artigo/*[local-name(.)="primeiro_autor" or local-name(.)="co-autor"]`
Artigo {primeiro_autor} Autor	`primeiro_autor`
Autor {primeiro_autor} Artigo	`[local-name(.)="primeiro_autor"]/..`
Artigo {co_autor} Autor	`co-autor`
Autor {co_autor}Artigo	`[local-name(.)="co-autor"]/..`

um elemento do esquema global retorna apenas um subconjunto dos resultados acessíveis através desse esquema, esse mapeamento possui uma suposição $sound$. Se a consulta *XPath* retorna exatamente o conjunto de resultados acessíveis através do esquema global, a suposição do mapeamento é $exact$. Se a consulta *XPath* retorna um superconjunto dos resultados, a suposição do mapeamento é $complete$.

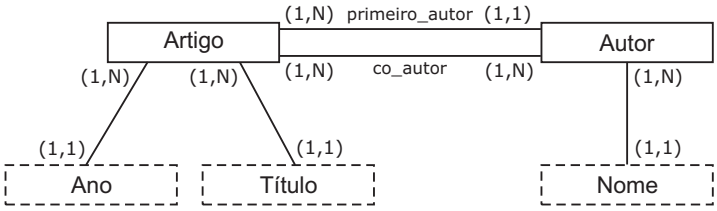


Figura 2.7: Modelo conceitual com relacionamentos identificados.

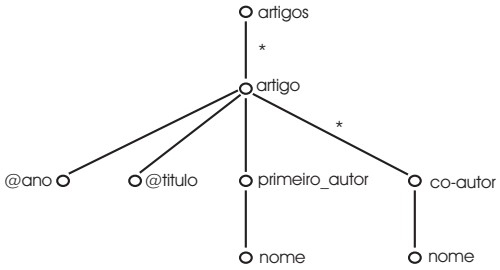


Figura 2.8: Esquema XML.

Exemplo 2.4 *Considere a Figura 2.9. O conceito $GlobalConcept$ representa um conceito do esquema global possuindo as instâncias* `{a,b,c}`*. Esse conceito está relacionado aos elementos $LocalConcept_i$ das fontes através de mapeamentos q_i, onde i representa a fonte. Sendo assim, quando o mapeamento q_1 é aplicado à fonte 1, o resultado seria composto pelas instâncias* `{a,c}`*. Pode-se notar que o mapeamento q_1, bem como o mapeamento q_3, ambos retornam um subconjunto das instâncias do conceito $GlobalConcept$. Por essa razão, é dito que tais mapeamentos são do tipo $sound$. Já o mapeamento q_2 retorna exatamente o conjunto de instâncias do conceito $GlobalConcept$, sendo assim, é considerado $exact$. Por fim, o mapeamento q_4 é considerado $complete$ pois retorna um superconjunto das instâncias de $GlobalConcept$.*

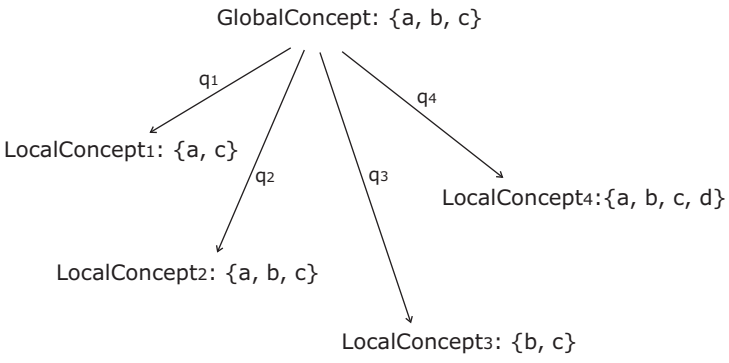


Figura 2.9: Suposições dos mapeamentos.

2.5 Arcabouço Formal para Integração de Dados

Esta seção tem como objetivo apresentar o arcabouço formal para integração de dados

proposto por (LENZERINI, 2002). Este arcabouço é bastante difundido na área de integração de dados e utilizado em diversos trabalhos (POGGI; ABITEBOUL, 2005; CALÌ et al., 2003; CALÌ et al., 2004; CALVANESE et al., 2001; LOPATENKO, 2004; EITER, 2005; LEONE et al., 2005). De acordo com o próprio autor, tal arcabouço é geral o bastante para capturar todas as abordagens de integração encontradas na literatura, o que inclui a abordagem para integração de dados utilizada nesta dissertação.

2.5.1 Arcabouço Lógico

Os principais componentes de um sistema de integração de dados são o esquema global, as fontes de dados e os mapeamentos. Formalmente, um sistema de integração de dados $\mathcal{I}$ é definido em termos de uma tripla $\langle \mathcal{G}, \mathcal{S}, \mathcal{M} \rangle$, onde:

- $\mathcal{G}$ é o *esquema global*, expresso em uma linguagem $\mathcal{L}_\mathcal{G}$ sob um alfabeto $\mathcal{A}_\mathcal{G}$. O alfabeto consiste de um símbolo para cada elemento de $\mathcal{G}$ (por exemplo, uma relação se $\mathcal{G}$ é relacional, um conceito ou papel se $\mathcal{G}$ é descrito em Lógica Descritiva, um rótulo se $\mathcal{G}$ é uma DTD, etc.) .

- $\mathcal{S} = \{S_1, S_2, ..., S_n\}$ é o conjunto de *esquemas das fontes*, onde cada esquema $\mathcal{S}_i$ é expresso através de uma linguagem $\mathcal{L}_\mathcal{S}$ sob um alfabeto $\mathcal{A}_\mathcal{S}$. O alfabeto $\mathcal{A}_\mathcal{S}$ inclui um símbolo para cada elemento das fontes (similar ao esquema global).

- $\mathcal{M} = \{M_1, M_2, ..., M_n\}$ contém *informações de mapeamento* relacionando elementos do esquema global $\mathcal{G}$ com elementos das fontes $\mathcal{S}$. Cada M_i por sua vez consiste em um conjunto de assertivas especificadas para a fonte i, cada qual com a forma:

  $$\langle g, q_\mathcal{S}, as \rangle \text{ para mapeamentos GAV, ou}$$
  $$\langle s, q_\mathcal{G}, as \rangle \text{ para mapeamentos LAV}$$

  onde g e s são elementos do esquema global e das fontes, respectivamente, e $q_\mathcal{G}$ e $q_\mathcal{S}$ são duas consultas respectivamente sobre o esquema global $\mathcal{G}$, e sobre os esquemas das fontes $\mathcal{S}$. Cada assertiva possui um componente as que representa a precisão do mapeamento, podendo assumir os valores $sound$, $exact$ ou $complete$. Consultas $q_\mathcal{S}$ são expressas através de uma linguagem de consulta $\mathcal{L}_{\mathcal{M},\mathcal{S}}$ sob o alfabeto $\mathcal{A}_\mathcal{S}$, e consultas $q_\mathcal{G}$ são expressas através de uma linguagem de consulta $\mathcal{L}_{\mathcal{M},\mathcal{G}}$ sob o alfabeto $\mathcal{A}_\mathcal{G}$.

Essa dissertação utiliza a abordagem GAV para definição dos mapeamentos (os mapeamentos GAV e LAV, assim como suas precisões, são explicados mais adiante nesta seção).

Afim de determinar a semântica de um sistema de integração de dados $\mathcal{I} = \langle \mathcal{G}, \mathcal{S}, \mathcal{M} \rangle$ considere um conjunto de fontes $\mathcal{D} = \{D_1, D_2, ..., D_n\}$ onde cada fonte D_i está de acordo com o esquema S_i pertencente ao conjunto de esquemas $\mathcal{S} = \{S_1, S_2, ..., S_n\}$. Baseado em $\mathcal{D}$ é especificado qual o conteúdo do esquema global $\mathcal{G}$.

Definição 2.5 *Chama-se de banco de dados global para um sistema de integração $\mathcal{I} = \langle \mathcal{G}, \mathcal{S}, \mathcal{M} \rangle$ qualquer instância do esquema global $\mathcal{G}$, ou seja, qualquer banco de dados que atende as restrições impostas pelo esquema global.*

Com essa definição é possível definir a semântica de um sistema de integração de dados.

Definição 2.6 *Dado um conjunto de fontes $\mathcal{D}$ de acordo com $\mathcal{S}$, chama-se o conjunto de bancos de dados globais válidos para $\mathcal{I} = \langle \mathcal{G}, \mathcal{S}, \mathcal{M} \rangle$ em relação a $\mathcal{D}$, denominado $sem(\mathcal{I}, \mathcal{D})$, o conjunto de bancos de dados $\mathcal{B}$ tais que:*

- *$\mathcal{B}$ é um banco de dados global para $\mathcal{I}$;*
- *$\mathcal{B}$ satisfaz os mapeamentos $\mathcal{M}$ em relação a $\mathcal{D}$.*

A noção de $\mathcal{B}$ satisfazer os mapeamentos $\mathcal{M}$ em relação a $\mathcal{D}$ depende da semântica das assertivas de cada mapeamento. Intuitivamente, a assertiva $\langle g, q_{\mathcal{S}}, as \rangle$ significa que o conceito representado por g do esquema global corresponde ao elemento da fonte representado pela consulta $q_{\mathcal{S}}$, com a precisão especificada por as (similar para $\langle s, q_{\mathcal{G}}, as \rangle$). Formalmente, seja q uma consulta e $\mathcal{DB}$ um banco de dados, é denotado por $q^{\mathcal{DB}}$ o conjunto de elementos em $\mathcal{DB}$ que satisfaz a consulta q. Com isso, dado um conjunto de fontes $\mathcal{D}$ de acordo com $\mathcal{S}$, e um banco de dados global $\mathcal{B}$, é dito que $\mathcal{B}$ satisfaz $\mathcal{M}$ em relação a $\mathcal{D}$, se para cada assertiva $\langle g, q_{\mathcal{S}}, as \rangle$ em $M_i \in \mathcal{M}$ tem-se que:

- se $as = sound$, então $g^{\mathcal{B}} \supseteq q_{\mathcal{S}}^{\mathcal{D}}$
- se $as = complete$, então $g^{\mathcal{B}} \subseteq q_{\mathcal{S}}^{\mathcal{D}}$
- se $as = exact$, então $g^{\mathcal{B}} = q_{\mathcal{S}}^{\mathcal{D}}$

Quando a precisão do mapeamento é especificada como $sound$ tem-se que o resultado $q_{\mathcal{S}}^{\mathcal{D}}$ da consulta $q_{\mathcal{S}}$ sobre o conjunto de fontes $\mathcal{D}$ está contido no conjunto de instâncias $g^{\mathcal{B}}$ do elemento g no banco de dados global $\mathcal{B}$ (similar para as precisões $complete$ e $exact$).

As abordagens encontradas na literatura para especificação dos mapeamentos entre o esquema global $\mathcal{G}$ e as fontes $\mathcal{S}$ são:

- *Local-As-View* (LAV) (ULLMAN, 2000): o esquema global é criado independente das fontes, e os relacionamentos entre o esquema global e as fontes são estabelecidos definindo cada fonte como uma visão do esquema global. Essa abordagem é bastante utilizada quando existe um esquema global estável e bem definido porque favorece a extensibilidade do sistema de integração, pois adicionando uma nova fonte significa apenas enriquecer os mapeamentos com uma nova assertiva, sem outras mudanças. Os mapeamentos em $\mathcal{M}$ têm a forma $\langle s, q_{\mathcal{G}}, as \rangle$, onde s é um elemento da fonte;
- *Global-As-View* (GAV) (HALEVY, 2001): o esquema global é criado a partir das fontes, ou seja, como uma visão sobre as fontes de dados. Para cada elemento do esquema global é associada uma visão sobre as fontes, tornando essa abordagem mais procedural, pois cada mapeamento especifica exatamente como extrair os dados das fontes. Os mapeamentos em $\mathcal{M}$ têm a forma $\langle g, q_{\mathcal{S}}, as \rangle$, onde g é um elemento do esquema global;
- *Global-And-Local-As-View* (GLAV) (LENZERINI, 2002): possibilita a especificação tanto de mapeamentos GAV quanto LAV explorando assim as vantagens de ambas as abordagens. Não existe um formato particular para esta abordagem.

Esta dissertação considera apenas mapeamentos do tipo GAV (global-as-view) com a precisão especificada como $sound$. A utilização de apenas mapeamentos GAV se dá por causa utilização da abordagem BInXS, que representa cada elemento do modelo conceitual através de uma visão *XPath* sobre as fontes. A precisão $sound$ é considerada pois é levado em consideração que as fontes de dados a serem integradas sejam provenientes da Web, sendo assim, não necessariamente tais fontes possuem todos os dados de interesse quando acessadas a partir do esquema global. Essa é a suposição da maioria dos trabalhos em integração de dados.

2.5.2 Resolução de Consultas em Sistemas de Integração de Dados

O serviço básico de um sistema de integração de dados é responder consultas, ou seja, a habilidade de responder consultas realizadas em termos do esquema global $\mathcal{G}$ e expressas em uma linguagem $\mathcal{L}_q$ sob um alfabeto $\mathcal{A}_\mathcal{G}$.

Dado um esquema global $\mathcal{G}$ e as fontes $\mathcal{D}$ envolvidas no sistema de integração, podem existir vários bancos de dados globais válidos possíveis. Isso torna a tarefa de resolver consultas mais complicada, o que requer a introdução da noção de *certain answers*.

Dado um sistema de integração de dados $\mathcal{I} = \langle \mathcal{G}, \mathcal{S}, \mathcal{M} \rangle$ e um conjunto de fontes $\mathcal{D}$ de acordo com os esquemas das fontes $\mathcal{S}$, as respostas corretas (*certain answers*), $q(\mathcal{I}, \mathcal{D})$, de uma consulta global q submetida à $\mathcal{I}$ em relação à $\mathcal{D}$, é o conjunto de tuplas t de elementos do domínio Γ (por exemplo, o domínio de instâncias de $\mathcal{G}$) tal que $t \in q^\mathcal{B}$ para todo banco de dados válido $\mathcal{B}$ em relação à $\mathcal{I}$:

$$q(\mathcal{I}, \mathcal{D}) = \{t \mid t \in q^\mathcal{B}, \forall \mathcal{B} \in sem(\mathcal{I}, \mathcal{D})\}$$

O problema da resolução de consultas pode ser tratado de duas formas diferentes. Em particular, a forma chamada *recognition* é formulada como segue:

1. Dado um sistema de integração $\mathcal{I} = \langle \mathcal{G}, \mathcal{S}, \mathcal{M} \rangle$, um conjunto de fontes $\mathcal{D}$ respeitando $\mathcal{S}$, uma consulta q, e uma tupla t de elementos do domínio Γ, verificar se t pertence ao resultado $q(\mathcal{I}, \mathcal{D})$;

Outras vezes, a resolução de consultas tem como meta encontrar o conjunto das *certain answers*. Então, o problema é formulado como a seguir:

2. Dado um sistema de integração $\mathcal{I} = \langle \mathcal{G}, \mathcal{S}, \mathcal{M} \rangle$, um conjunto de fontes $\mathcal{D}$ respeitando $\mathcal{S}$, e uma consulta q, encontrar todo t que pertence a $q(\mathcal{I}, \mathcal{D})$.

Por ser uma abordagem mais prática, esta dissertação trata o problema de resolver consultas de acordo com a segunda forma, ou seja, encontrar todas as respostas de uma consulta.

2.6 Arcabouço Formal para Integração de Dados XML baseada em Modelos Conceituais

Nesta seção, são reapresentados os principais componentes da abordagem para integração de dados utilizada neste trabalho, mostrados informalmente nas Seções 2.1-2.4, utilizando o arcabouço formal de (LENZERINI, 2002) apresentado na seção anterior. A leitura dos capítulos seguintes é independente dessa seção, exceto a Seção 4.6 que revisa a definição formal aqui apresentada para incluir dependências de inclusão no esquema global.

Definição 2.7 *Um modelo conceitual $\mathcal{CM}(Conceptual\ Model)$ é representado pela tupla $\mathcal{CM} = \langle \mathcal{NL}, \mathcal{L}, \mathcal{R} \rangle$, onde:*

- *$\mathcal{NL}$ é um conjunto de conceitos não-léxicos.*
- *$\mathcal{L}$ é um conjunto de conceitos léxicos.*
- *$\mathcal{R}$ é um conjunto de relacionamentos entre conceitos. Um relacionamento $r \in \mathcal{R}$ pode ser classificado como um relacionamento de associação ou um relacionamento de herança ($\mathcal{R} = \mathcal{R}_a \cup \mathcal{R}_h$):*
  - *Um relacionamento de associação $r \in \mathcal{R}_a$ é uma tupla $\langle id_1, id_2, c_d, c_i, n, p_1, p_2 \rangle$, onde:*
    - *id_1 e id_2 são conceitos, sendo $id_1 \in \mathcal{NL}$ e $id_2 \in \mathcal{NL} \cup \mathcal{L}$*
    - *c_d e c_i são as cardinalidades, onde c_d é a cardinalidade direta (de id_1 para id_2) e c_i é a cardinalidade inversa (de id_2 para id_1). Cada cardinalidade é representada por uma tupla $\langle min, max \rangle$, representando a cardinalidade mínima e máxima.*
    - *n é o nome do relacionamento (ε para relacionamentos sem nome).*
    - *p_1 e p_2 são os papéis assumidos no relacionamento n por id_1 e id_2 respectivamente.*
  - *Um relacionamento de herança $r_h \in \mathcal{R}_h$ é representado por $\langle id_g, id_e \rangle$, onde id_g representa o conceito genérico e id_e representa o conceito especializado, onde $id_g, id_e \in \mathcal{NL}$.*

No modelo conceitual, um conceito especial chamado $Root$ se distingue dos demais porque possui um relacionamento não identificado com todos os outros conceitos. Portanto, é sempre possível navegar a partir do conceito $Root$ para qualquer outro conceito dentro do modelo conceitual. Tal conceito é utilizado nesta dissertação de forma a uniformizar a definição dos mecanismos de tradução e de reescrita de consultas a serem vistos nos Capítulos 3 e 4, fazendo com que cada conceito possua um determinado contexto. A Figura 2.10 mostra como o conceito $Root$ se relaciona com os demais em um modelo conceitual.

Exemplo 2.8 *Considere o sistema de integração de dados $\mathcal{I} = \langle \mathcal{CM}, \mathcal{S}, \mathcal{M} \rangle$. O esquema global é mostrado na Figura 2.10, o conjunto $\mathcal{S}$ de esquemas XML é mostrado na Figura 2.11, e as informações de mapeamento $\mathcal{M}$ entre $\mathcal{CM}$ e $\mathcal{S}$ são apresentadas nas Figuras 2.12 e 2.13.*

■

O conceito $Root$ está sempre presente em qualquer modelo conceitual. Por simplicidade de notação, nos próximos exemplos tal conceito será omitido. As cardinalidades não têm influência sobre o processo de tradução dos Capítulos 3 e 4. Por esse motivo, as informações sobre as cardinalidades também serão omitidas.

Um sistema de integração de dados $\mathcal{I}$ é uma tripla $\langle \mathcal{CM}, \mathcal{S}, \mathcal{M} \rangle$, onde:

- $\mathcal{CM}$ é o esquema global, representado através de um modelo conceitual. A formalização do modelo conceitual é apresentada na Definição 2.7;

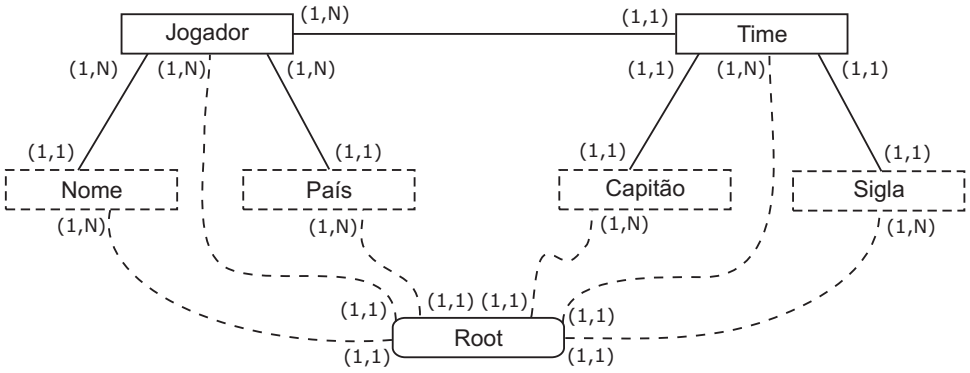
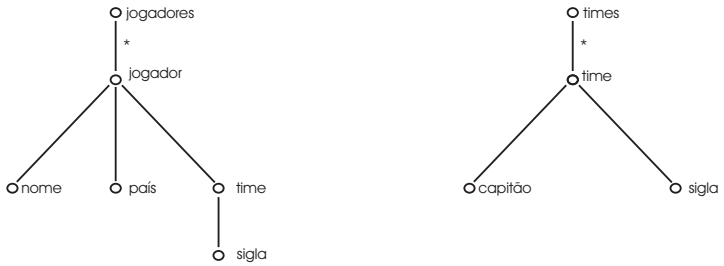


Figura 2.10: Modelo Conceitual formado a partir dos esquemas da Figura 2.11.



(a) Esquema XML 1 (b) Esquema XML 2

Figura 2.11: Esquemas XML utilizados para validação.

- $\mathcal{S}$ é o conjunto $\mathcal{S} = \{S_1, S_2, ..., S_n\}$ de esquemas XML das fontes;
- $\mathcal{M}$ é o conjunto $\{M_1, M_2, ..., M_n\}$ de mapeamentos especificados entre $\mathcal{CM}$ e $\mathcal{S}$, do tipo *global-as-view (GAV)*, onde cada mapeamento M_i, relativo a fonte i, é formado por um conjunto de tuplas $\langle g, q_{\mathcal{S}}, as \rangle$, onde: g é representado através de $Root\ \varepsilon\ id$ para todo conceito id do modelo conceitual, e representado através de $id_1\ Relationship\ id_2$ para todo relacionamento de nome $Relationship$ entre os conceitos id_1 e id_2 do modelo; $q_{\mathcal{S}}$ é uma consulta *XPath* sobre a fonte em questão; e a precisão as possui o valor $sound$[1].

Uma consulta submetida ao sistema de integração $\mathcal{I}$ (consultas dos usuários) é uma declaração que especifica quais os dados que devem ser retornados do sistema. Cada consulta do usuário submetida ao modelo conceitual $\mathcal{CM}$ é realizada utilizando a linguagem de consulta *CXPath*, cuja gramática e exemplos foram apresentados na Seção 2.3.

Afim de determinar a semântica de um sistema de integração de dados $\mathcal{I} = \langle \mathcal{CM}, \mathcal{S}, \mathcal{M} \rangle$ considere o conjunto de fontes XML $\mathcal{D} = \{D_1, D_2, ..., D_n\}$ de acordo com o conjunto de esquemas XML $\mathcal{S} = \{S_1, S_2, ..., S_n\}$. Um *banco de dados global* para $\mathcal{I}$ é qualquer instância do modelo conceitual $\mathcal{CM}$, ou seja, qualquer banco de dados que atende as restrições impostas pelo modelo conceitual. Uma definição formal de banco de dados global, apresentada no trabalho (SILVEIRA, 2006), é dada na Definição 2.9.

[1]Neste trabalho é assumido que todos os mapeamentos são definidos com a precisão $sound$, por esse motivo, nos exemplos que seguem, essa informação é omitida.

$$
\begin{aligned}
M_1 \;=\; \{\; & \langle \texttt{Root}\ \varepsilon\ \texttt{Jogador}, && \texttt{/jogadores/jogador}\rangle \\
& \langle \texttt{Root}\ \varepsilon\ \texttt{Time}, && \texttt{/jogadores/jogador/time}\rangle \\
& \langle \texttt{Root}\ \varepsilon\ \texttt{País} && \texttt{/jogadores/jogador/país}\rangle \\
& \langle \texttt{Root}\ \varepsilon\ \texttt{Nome}, && \texttt{/jogadores/jogador/nome}\rangle \\
& \langle \texttt{Root}\ \varepsilon\ \texttt{Sigla} && \texttt{/jogadores/jogador/time/sigla}\rangle \\
& \langle \texttt{Root}\ \varepsilon\ \texttt{Capitão}, && \texttt{indefinido}\,\rangle \\
& \langle \texttt{Jogador}\ \varepsilon\ \texttt{País}, && \texttt{país}\rangle \\
& \langle \texttt{Jogador}\ \varepsilon\ \texttt{Nome}, && \texttt{nome}\rangle \\
& \langle \texttt{Jogador}\ \varepsilon\ \texttt{Time}, && \texttt{time}\rangle \\
& \langle \texttt{Time}\ \varepsilon\ \texttt{Jogador}, && \texttt{..}\,\rangle \\
& \langle \texttt{Time}\ \varepsilon\ \texttt{Sigla}, && \texttt{sigla}\rangle \\
& \langle \texttt{Time}\ \varepsilon\ \texttt{Capitão}, && \texttt{indefinido}\,\rangle \\
& \langle \texttt{País}\ \varepsilon\ \texttt{Jogador}, && \texttt{..}\,\rangle \\
& \langle \texttt{Nome}\ \varepsilon\ \texttt{Jogador}, && \texttt{..}\,\rangle \\
& \langle \texttt{Sigla}\ \varepsilon\ \texttt{Time}, && \texttt{..}\,\rangle \\
& \langle \texttt{Capitão}\ \varepsilon\ \texttt{Time}, && \texttt{indefinido}\,\rangle \quad \}
\end{aligned}
$$

Figura 2.12: Informações de Mapeamento dos esquema XML 1 da Figura 2.11.

$$
\begin{aligned}
M_2 \;=\; \{\; & \langle \texttt{Root}\ \varepsilon\ \texttt{Jogador}, && \texttt{indefinido}\rangle \\
& \langle \texttt{Root}\ \varepsilon\ \texttt{Time}, && \texttt{/times/time}\rangle \\
& \langle \texttt{Root}\ \varepsilon\ \texttt{País} && \texttt{indefinido}\rangle \\
& \langle \texttt{Root}\ \varepsilon\ \texttt{Nome}, && \texttt{/times/time/capitão}\rangle \\
& \langle \texttt{Root}\ \varepsilon\ \texttt{Sigla} && \texttt{/times/time/sigla}\rangle \\
& \langle \texttt{Root}\ \varepsilon\ \texttt{Capitão}, && \texttt{/times/time/capitão}\,\rangle \\
& \langle \texttt{Jogador}\ \varepsilon\ \texttt{País}, && \texttt{indefinido}\rangle \\
& \langle \texttt{Jogador}\ \varepsilon\ \texttt{Nome}, && \texttt{indefinido}\rangle \\
& \langle \texttt{Jogador}\ \varepsilon\ \texttt{Time}, && \texttt{indefinido}\rangle \\
& \langle \texttt{Time}\ \varepsilon\ \texttt{Jogador}, && \texttt{indefinido}\,\rangle \\
& \langle \texttt{Time}\ \varepsilon\ \texttt{Sigla}, && \texttt{sigla}\rangle \\
& \langle \texttt{Time}\ \varepsilon\ \texttt{Capitão}, && \texttt{capitão}\,\rangle \\
& \langle \texttt{País}\ \varepsilon\ \texttt{Jogador}, && \texttt{indefinido}\,\rangle \\
& \langle \texttt{Nome}\ \varepsilon\ \texttt{Jogador}, && \texttt{indefinido}\,\rangle \\
& \langle \texttt{Sigla}\ \varepsilon\ \texttt{Time}, && \texttt{..}\,\rangle \\
& \langle \texttt{Capitão}\ \varepsilon\ \texttt{Time}, && \texttt{..}\,\rangle \quad \}
\end{aligned}
$$

Figura 2.13: Informações de Mapeamento dos esquema XML 2 da Figura 2.11.

Definição 2.9 *Uma base de dados $\mathcal{B}$ para um modelo conceitual $\mathcal{CM} = \langle \mathcal{NL}, \mathcal{L}, \mathcal{R} \rangle$ é dada por uma tupla $\mathcal{B} = \langle \mathcal{NL}_\mathcal{B}, \mathcal{L}_\mathcal{B}, \mathcal{R}_\mathcal{B} \rangle$, onde:*

- *$\mathcal{NL}_\mathcal{B}$ é o conjunto de instâncias não léxicas. Uma instância não-léxica é uma tupla $\langle oid, id \rangle$, onde oid é o identificador do objeto e $id \in \mathcal{NL}$;*
- *$\mathcal{L}_\mathcal{B}$ é o conjunto de instâncias léxicas. Uma instância léxica é uma tupla $\langle id, v \rangle$, onde $id \in \mathcal{L}$, e v é o valor do conceito;*
- *$\mathcal{R}_{a\mathcal{B}}$ é o conjunto de instâncias de relacionamentos de associação. Uma instância desse tipo de relacionamento é uma tupla $\langle i_1, i_2 \rangle$, onde $i_1 \in \mathcal{NL}_\mathcal{B}$, $i_2 \in \mathcal{NL}_\mathcal{B} \cup \mathcal{L}_\mathcal{B}$, tal que existe um relacionamento de associação entre $i_1.id$ e $i_2.id$ em $\mathcal{R}_a$;*
- *$\mathcal{R}_{h\mathcal{B}}$ é o conjunto de instâncias de relacionamentos de herança. Uma instância desse tipo de relacionamento é uma tupla $\langle i_1, i_2 \rangle$, onde $i_1, i_2 \in \mathcal{NL}_\mathcal{B}$ e existe um relacionamento de herança entre $i_1.id$ e $i_2.id$ em $\mathcal{R}_h$ no qual i_1 é o conceito genérico e i_2 é o conceito especializado;*
- *A base de dados $\mathcal{CM}_\mathcal{B}$ obedece às restrições de cardinalidade definidas no modelo conceitual $\mathcal{CM}$.*

Dado um conjunto $\mathcal{D}$ de fontes XML de acordo com um conjunto de esquemas XML $\mathcal{S}$, chama-se o conjunto de *bancos de dados válidos* para $\mathcal{I}$ em relação a $\mathcal{D}$, denominado $sem(\mathcal{I}, \mathcal{D})$, o conjunto de bancos de dados $\mathcal{B}$ tais que:

- $\mathcal{B}$ é um banco de dados global;
- $\mathcal{B}$ satisfaz os mapeamentos $\mathcal{M}$ em relação a $\mathcal{D}$.

A assertiva $\langle g, q_\mathcal{S} \rangle$ significa que o elemento g representado no modelo conceitual corresponde ao conceito do esquema da fonte representado pela consulta *XPath* $q_\mathcal{S}$, com a precisão definida como $sound$. Formalmente, dado um conjunto de fontes $\mathcal{D}$ de acordo com $\mathcal{S}$ e um banco de dados global $\mathcal{B}$, é dito que $\mathcal{B}$ satisfaz os mapeamentos $\mathcal{M}$ do tipo $sound$, em relação a $\mathcal{D}$, se para cada $M_i \in \mathcal{M}$ cada tupla $\langle g, q_\mathcal{S} \rangle$ segue o padrão $g^\mathcal{B} \supseteq q_\mathcal{S}^\mathcal{D}$.

Isso confirma a precisão $sound$ dos mapeamentos utilizada nesta dissertação, onde o elemento do esquema XML representado por $q_\mathcal{S}$ quando aplicado à fonte $D_i \in \mathcal{D}$ retorna um subconjunto dos elementos pertencentes à extensão do elemento g do esquema global dado um banco de dados $\mathcal{B} \in sem(\mathcal{I}, \mathcal{D})$.

Dado um sistema de integração de dados $\mathcal{I} = \langle \mathcal{CM}, \mathcal{S}, \mathcal{M} \rangle$ e um conjunto de fontes $\mathcal{D}$ de acordo com $\mathcal{S}$, as respostas corretas (*certain answers*), $\textit{CXPath}^{\mathcal{I},\mathcal{D}}$, de uma consulta *CXPath* submetida à $\mathcal{I}$ em relação à $\mathcal{D}$, é o conjunto de objetos t de elementos do domínio Γ (por exemplo, o domínio de instâncias de $\mathcal{G}$) tal que $t \in \textit{CXPath}^\mathcal{B}$ a todo banco de dados válido $\mathcal{B}$ para o sistema de integração $\mathcal{I}$ em relação à $\mathcal{D}$:

$$\textit{CXPath}^{\mathcal{I},\mathcal{D}} = \{t \mid t \in \textit{CXPath}^\mathcal{B}, \forall \mathcal{B} \in sem(\mathcal{I}, \mathcal{D})\}$$

3 MECANISMO DE TRADUÇÃO DE CONSULTAS CXPATH PARA CONSULTAS XPATH

O objetivo deste capítulo é formalizar o mecanismo de tradução de consultas *CXPath* para consultas *XPath*. São apresentados também alguns exemplos de tradução de consultas *CXPath* para consultas *XPath*. É mostrado o tratamento do relacionamento de herança no mecanismo de tradução, e por fim, é feita uma discussão sobre a relação entre a expressividade do modelo conceitual e o mecanismo de tradução.

Em (CAMILLO; HEUSER; MELLO, 2003), cada fonte de dados é tratada de forma independente, ou seja, não há interação entre as fontes afim de construir o resultado de uma consulta. No mecanismo de tradução apresentado neste capítulo é utilizada essa mesma suposição, ou seja, não é tratado o problema da fragmentação dos dados no mecanismo de tradução proposto.

3.1 Descrição do Mecanismo de Tradução

Na abordagem utilizada neste trabalho para integração de dados é utilizado um modelo conceitual como esquema global afim de abstrair fontes XML (MELLO; HEUSER, 2005). Para tornar possível a realização de consultas sob o modelo conceitual foi criada a linguagem *CXPath (Conceptual XPath)* (CAMILLO; HEUSER; MELLO, 2003). A sintaxe da linguagem *CXPath* pode ser vista na Figura 3.1.

Dada essa abordagem, é necessário traduzir consultas em nível conceitual para consultas em nível XML. Consultas *CXPath* submetidas ao modelo conceitual devem ser traduzidas, com base nas informações de mapeamento, para consultas *XPath* afim de extrair os dados nas fontes XML.

Alguns trabalhos utilizam um único modelo de dados tanto no nível global quanto no nível local, facilitando assim a interação entre o esquema global e as fontes de dados. Por exemplo, em (CALÌ et al., 2001) é utilizado o modelo relacional tanto para o esquema global quanto para as fontes. Já em (POGGI; ABITEBOUL, 2005), é utilizado o modelo XML no esquema global e nas fontes.

O mecanismo de tradução de consultas *CXPath* em consultas *XPath* consiste em reescrever cada conceito e relacionamento da consulta *CXPath* pela informação de mapeamento correspondente na fonte XML em questão. A consulta *CXPath*, dada como entrada, é avaliada da esquerda para direita, e traduzida para cada fonte XML relevante. As regras para tradução de consultas *CXPath* em consultas *XPath* são apresentadas na próxima seção.

$$
\begin{array}{lcl}
CXPath & ::= & AbsolutePath \\
 & | & RelativePath \\
 & & \\
AbsolutePath & ::= & / \\
 & | & / \ RelativePath \\
 & & \\
RelativePath & ::= & Relationship \ id \ Predicates \\
 & | & Relationship \ id \ Predicates \ / \ RelativePath \\
 & & \\
Relationship & ::= & \{ \ RelationshipName \ \} \\
 & | & \{ \ RelationshipName.RoleName \ \} \\
 & | & \varepsilon \\
 & & \\
Predicates & ::= & [/RelativePath_1 \ op \ /RelativePath_2] \ Predicates \\
 & | & [/RelativePath_1 \ op \ RelativePath_2] \ Predicates \\
 & | & [/RelativePath \ op \ Literal] \ Predicates \\
 & | & [RelativePath_1 \ op \ /RelativePath_2] \ Predicates \\
 & | & [RelativePath_1 \ op \ RelativePath_2] \ Predicates \\
 & | & [RelativePath \ op \ Literal] \ Predicates \\
 & | & \varepsilon \\
 & & \\
Literal & ::= & IntegerLiteral \\
 & | & StringLiteral \\
 & & \\
op & ::= & = \ | \ != \ | \ < \ | \ <= \ | \ > \ | \ >=
\end{array}
$$

Figura 3.1: Gramática de *CXPath*.

3.2 Formalização do Mecanismo de Tradução

As regras de tradução de consultas *CXPath* para consultas *XPath* são definidas indutivamente sobre a gramática de *CXPath* (Figura 3.1). Dada uma consulta *CXPath* e informações de mapeamento $\mathcal{M}$ o mecanismo de tradução produz uma consulta *XPath* para cada fonte relevante:

$$\mathsf{Tr} : \mathsf{CXPath} \times \mathcal{M} \rightarrow \mathsf{XPath}$$

O mecanismo de tradução Tr é composto pelas funções $\mathbf{Tr_{APE}}$, $\mathbf{Tr_{RPE}}$, $\mathbf{Tr_{REL}}$, e $\mathbf{Tr_{PRE}}$. Elas são responsáveis pela tradução respectivamente, das expressões de caminho absoluto, expressões de caminho relativo, relacionamentos e predicados.

No que segue são apresentadas as regras para tradução de consultas *CXPath* em consultas *XPath*. As informações de mapeamento $\mathcal{M}$ são acessadas somente pela função $\mathbf{Tr_{REL}}$ (Subseção 3.2.3) para tradução de relacionamentos. Por esse motivo, e por questão de legibilidade, o argumento correspondente a essas informações de mapeamento é omitido na definição das demais funções.

3.2.1 Tradução de Expressões de Caminho Absoluto ($\mathbf{Tr_{APE}}$)

Existem duas regras de tradução para expressões absolutas, uma para cada cláusula $AbsolutePath$ da gramática da Figura 3.1 ("/" e "$/RelativePath$"). Pela regra (TRAPE1) a expressão de caminho absoluta "/" é traduzida para a própria "/".

(TRAPE1) $\mathbf{Tr_{APE}}(/) = /$

Pela regra (TRAPE2), a tradução de uma expressão de caminho absoluta com a forma "$/RelativePath$" fica a cargo da função de tradução de caminho relativo $\mathbf{Tr_{RPE}}$ que possui dois argumentos: o conceito de referência, ou seja, o contexto de avaliação, e o caminho relativo à esse contexto. A função $\mathbf{Tr_{RPE}}$ é definida na Subseção 3.2.2.

(TRAPE2) $\mathbf{Tr_{APE}}(/RelativePath) = \mathbf{Tr_{RPE}}(Root, RelativePath)$

Nas demais funções esse será um padrão recorrente, ou seja, a tradução sempre é feita em relação a algum contexto passado como argumento.

3.2.2 Tradução de Expressões de Caminho Relativo ($\mathbf{Tr_{RPE}}$)

Há duas cláusulas na gramática da Figura 3.1 para expressões de caminho relativas. A função $\mathbf{Tr_{RPE}}$, pela regra (TRRPE1) , quando recebe um contexto id_1 e uma expressão de caminho relativa $Relationship\ id_2\ Predicates$, traduz a informação $(id_1\ Relationship\, id_2)$ de acordo com a função $\mathbf{Tr_{REL}}$ (Subseção 3.2.3) para a fonte i, juntamente com a tradução dos predicados da expressão de caminho relativa, realizada pela função de tradução de predicados $\mathbf{Tr_{PRE}}$ (Subseção 3.2.4).

(TRRPE1) $\mathbf{Tr_{RPE}}(id_1,\ Relationship\ id_2\ \ Predicates) =$
$\mathbf{Tr_{REL}}(id_1\ \ Relationship\ \ id_2\ )\ \ \mathbf{Tr_{PRE}}(id_2, Predicates)$

Quando existem outras expressões de caminho seguindo a primeira, a próxima expressão de caminho deve ser traduzida levando em consideração um determinado contexto, que é dado pela expressão anterior.

(TRRPE2) $\mathbf{Tr_{RPE}}(id_1,\ Relationship\ id_2\ \ Predicates\ /\ RelativePath) =$
$\mathbf{Tr_{REL}}(id_1\ \ Relationship\ \ id_2)\ \ \mathbf{Tr_{PRE}}(id_2, Predicates)\ \ /\ \ \mathbf{Tr_{RPE}}(id_2, RelativePath)$

3.2.3 Tradução de Relacionamentos ($\mathbf{Tr_{REL}}$)

A tradução dos relacionamentos nada mais é do que acessar a informação de mapeamento representada por $\langle id_1\ Relationship\ id_2,\ q_S \rangle \in M_i$. A informação necessária nesse processo de tradução é a consulta q_S que está associada ao elemento $id_1\ Relationship\ id_2$ de uma determinada fonte. A chamada M_i**(**$id_1\ Relationship\ id_2$**)** realizada sob a fonte i, retorna q_S associado com $id_1\ Relationship\ id_2$.

(TRREL) $\mathbf{Tr_{REL}}$**(**$id_1\ Relationship\ id_2$**)** $= M_i$**(**$id_1\ Relationship\ id_2$**)**

3.2.4 Tradução de Predicados ($\mathbf{Tr_{PRE}}$)

A tradução dos predicados nada mais é do que a aplicação de regras definidas acima. As regras para os predicados conservam os colchetes **[ ]**, os operadores op e os literais especificados na gramática de *CXPath*. A regra (TRPRE1) trata de predicados com operações entre expressões de caminho absolutas, enquanto que a regra (TRPRE5) trata de predicados com operações entre expressões relativas.

(TRPRE1) $\mathbf{Tr_{PRE}}$**(**$id,\ [/RelativePath_1\ \ op\ \ /RelativePath_2]\ Predicates$**)** $=$
$[\ \mathbf{Tr_{APE}}(/RelativePath_1)\ \ op\ \ \mathbf{Tr_{APE}}(/RelativePath_2)\ ]\ \mathbf{Tr_{PRE}}(id,\ Predicates)$

(TRPRE2) $\mathbf{Tr_{PRE}}$**(**$id,\ [RelativePath_1\ \ op\ \ RelativePath_2]\ Predicates$**)** $=$
$[\ \mathbf{Tr_{RPE}}(id, RelativePath_1)\ \ op\ \ \mathbf{Tr_{RPE}}(id, RelativePath_2)\ ]\ \mathbf{Tr_{PRE}}(id,\ Predicates)$

As regras (TRPRE2) e (TRPRE4) traduzem predicados com operações entre expressões de caminho absolutas e relativas. As expressões absolutas são traduzidas pela função $\mathbf{Tr_{APE}}$ e as expressões relativas são traduzidas pela função $\mathbf{Tr_{RPE}}$ utilizando o contexto o conceito id.

(TRPRE3) $\mathbf{Tr_{PRE}}$**(**$id,\ [/RelativePath_1\ \ op\ \ RelativePath_2]\ Predicates$**)** $=$
$[\ \mathbf{Tr_{APE}}(/RelativePath_1)\ \ op\ \ \mathbf{Tr_{RPE}}(id,\ RelativePath_2)\ ]\ \mathbf{Tr_{PRE}}(id,\ Predicates)$

(TRPRE4) $\mathbf{Tr_{PRE}}$**(**$id,\ [RelativePath_1\ \ op\ \ /RelativePath_2]\ Predicates$**)** $=$
$[\ \mathbf{Tr_{RPE}}(id,\ RelativePath_1)\ \ op\ \ \mathbf{Tr_{APE}}(/RelativePath_2)\ ]\ \mathbf{Tr_{PRE}}(id,\ Predicates)$

As regras (TRPRE3) e (TRPRE6) tratam de operações entre expressões de caminho, absolutas e relativas respectivamente, e literais, que podem ser uma $string$ ou um $inteiro$.

(TRPRE5) $\mathbf{Tr_{PRE}}$**(**$id,\ [/RelativePath_1\ \ op\ \ Literal]\ Predicates$**)** $=$
$[\ \mathbf{Tr_{APE}}(/RelativePath_1)\ \ op\ \ Literal\ ]\ \mathbf{Tr_{PRE}}(id,\ Predicates)$

(TRPRE6) $\mathbf{Tr_{PRE}}$**(**$id,\ [RelativePath_1\ \ op\ \ Literal]\ Predicates$**)** $=$
$[\ \mathbf{Tr_{RPE}}(id,\ RelativePath_1)\ \ op\ \ Literal\ ]\ \mathbf{Tr_{PRE}}(id,\ Predicates)$

Por fim, a regra (TRPRE7) trata de predicados vazios.

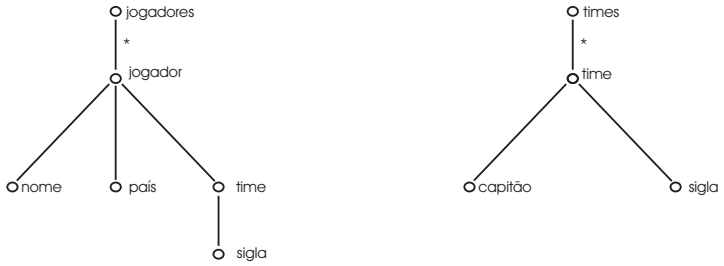
(TRPRE7) $\mathbf{Tr_{PRE}}$**(**$id,\ \varepsilon$**)** $= \varepsilon$

3.3 Exemplos de Tradução

Nesta seção são apresentados alguns exemplos de consultas *CXPath* e como elas são traduzidas para consultas *XPath* utilizando as funções de tradução apresentadas na Seção 3.2. O domínio de jogadores e times será utilizado: cada jogador possui um nome e um país de origem, e cada time possui uma sigla e um capitão.

O modelo conceitual utilizado nos exemplos a seguir é apresentado na Figura 3.2 criado a partir dos esquemas XML da Figura 3.3. As informações de mapeamento M_1 e M_2 são mostradas através das Tabelas 3.1 e 3.2, respectivamente dos esquemas XML 1 e 2 da Figura 3.3.

Figura 3.2: Modelo Conceitual dos esquemas da Figura 3.3.



(a) Esquema XML 1 (b) Esquema XML 2

Figura 3.3: Esquemas XML utilizados nos exemplos.

Exemplo 3.1 *Supor a consulta global CXPath* `/Jogador`*, cujo objetivo é obter todas as instâncias do conceito* $Jogador$*. Considere o processo de tradução dessa consulta CXPath para uma consulta XPath na* ***fonte XML 1****, cujo esquema é apresentado na Figura 3.3(a).*

Para expressões absolutas existem duas regras: (TRAPE1) *quando se trata apenas de uma "/" e* (TRAPE2) *quando se trata de uma "/" seguida de uma expressão relativa. Pode-se ver claramente que a regra* (TRAPE2) *será utilizada para traduzir a consulta,* `/Jogador`*. Como se trata de uma expressão absoluta, ela é traduzida utilizando o conjunto de regras para expressões relativas utilizando o conceito* $Root$ *como contexto. Portanto, pela regra* (TRAPE2)*,*

$$\mathbf{Tr_{APE}}(\texttt{/Jogador}) = \mathbf{Tr_{RPE}}(\texttt{Root}, \varepsilon\ \texttt{Jogador}\ \varepsilon) \quad (1)$$

Para expressões de caminho relativas há duas opções: regras (TRRPE1) *e* (TRRPE2)*. Como não existem outras expressões relativas de caminho seguindo a expressão*

Tabela 3.1: Informações de Mapeamento M_1.

Conceito/Relacionamento	Fonte XML 1
Root ε Jogador	`/jogadores/jogador`
Root ε Time	`/jogadores/jogador/time`
Root ε País	`/jogadores/jogador/país`
Root ε Nome	`/jogadores/jogador/nome`
Root ε Sigla	`/jogadores/jogador/time/sigla`
Root ε Capitão	`indefinido`
Jogador ε País	`país`
Jogador ε Nome	`nome`
Jogador ε Time	`time`
Time ε Jogador	..
Time ε Sigla	`sigla`
Time ε Capitão	`indefinido`
País ε Jogador	..
Nome ε Jogador	..
Sigla ε Time	..
Capitão ε Time	`indefinido`

Tabela 3.2: Informações de Mapeamento M_2.

Conceito/Relacionamento	Fonte XML 2
Root ε Jogador	`indefinido`
Root ε Time	`/times/time`
Root ε País	`indefinido`
Root ε Nome	`/times/time/capitão`
Root ε Sigla	`/times/time/sigla`
Root ε Capitão	`/times/time/capitão`
Jogador ε País	`indefinido`
Jogador ε Nome	`indefinido`
Jogador ε Time	`indefinido`
Time ε Jogador	`indefinido`
Time ε Sigla	`sigla`
Time ε Capitão	`capitão`
País ε Jogador	`indefinido`
Nome ε Jogador	`indefinido`
Sigla ε Time	..
Capitão ε Time	..

relativa `Jogador` *a regra utilizada neste momento da tradução é a regra* (TRRPE1). *Usando essa regra no lado direito de (1) obtém-se:*

$$\mathbf{Tr_{APE}}(\texttt{/Jogador}) = \mathbf{Tr_{REL}}(\texttt{Root } \varepsilon \texttt{ Jogador})\ \mathbf{Tr_{PRE}}(\texttt{Jogador}, \varepsilon) \qquad (2)$$

Para a tradução dos relacionamentos é utilizada a regra (TRREL). *Nesse momento é buscado nos mapeamentos para a fonte XML 1 a informação* $\mathbf{M_1}$(`Root` ε `Jogador`),

que retorna a consulta XPath para tal fonte. Utilizando a regra (TRREL) *no lado direito de (2) é tem-se:*

$$\mathbf{Tr_{APE}}(\texttt{/Jogador}) = M_1(\texttt{Root}\ \varepsilon\ \texttt{Jogador})\ \mathbf{Tr_{PRE}}(\texttt{Root}, \varepsilon) \tag{3}$$

Das regras especificadas para tratar predicados, a única que trata de predicados vazios é a regra (TRPRE7). *Aplicando essa regra em (3) obtém-se:*

$$\mathbf{Tr_{APE}}(\texttt{/Jogador}) = M_1(\texttt{Root}\ \varepsilon\ \texttt{Jogador})\ \varepsilon \tag{4}$$

Por fim, é retornado o resultado M_1(`Root` ε `Jogador`). *Com isso, é obtida a consulta XPath traduzida da consulta CXPath* `/Jogador` *de acordo com a Tabela 3.1:*

$$\mathbf{Tr_{APE}}(\texttt{/Jogador}) = \texttt{/jogadores/jogador}$$

■

Exemplo 3.2 *Neste exemplo será utilizada a mesma consulta CXPath* `/Jogador` *para tradução em relação a fonte XML 2, especificada na Figura 3.3(b). A ordem da aplicação das regras é similar à ordem do exemplo anterior pois se trata da mesma consulta CXPath.*

$$\begin{aligned}
\mathbf{Tr_{APE}}(\texttt{/Jogador}) &= \mathbf{Tr_{RPE}}(\texttt{Root}, \varepsilon\ \texttt{Jogador}\ \varepsilon) && \text{(TRAPE2)}\\
&= \mathbf{Tr_{REL}}(\texttt{Root}\ \varepsilon\ \texttt{Jogador})\ \mathbf{Tr_{PRE}}(\texttt{Root}, \varepsilon) && \text{(TRRPE1)}\\
&= M_2(\texttt{Root}\ \varepsilon\ \texttt{Jogador})\ \mathbf{Tr_{PRE}}(\texttt{Root}, \varepsilon) && \text{(TRREL)}\\
&= M_2(\texttt{Root}\ \varepsilon\ \texttt{Jogador})\ \varepsilon && \text{(TRPRE7)}\\
&= \texttt{indefinido}
\end{aligned}$$

Porém, $\mathbf{M_2}$(`Root` ε `Jogador`), *de acordo com as informações de mapeamento para a fonte 2 apresentadas na Tabela 3.2, retorna* `indefinido` *o que significa que não existe informações sobre jogadores nessa fonte, encerrando o processo de tradução.*

■

Exemplo 3.3 *Este exemplo mostra a tradução da consulta global CXPath* `/Time/Sigla` *afim de obter todas as instâncias de siglas de times presentes no sistema de integração. Essa consulta será traduzida para a fonte XML 1.*

Assim como no exemplo anterior, das duas regras existentes para expressões de caminho absolutas será utilizada a regra (TRAPE2) *usando o conceito* `Root` *como contexto:*

$$\mathbf{Tr_{APE}}(\texttt{/Time/Sigla}) = \mathbf{Tr_{RPE}}(\texttt{Root}, \varepsilon\ \texttt{Time}\ \varepsilon\ \texttt{/Sigla}) \tag{1}$$

Dessa vez, a expressão relativa `Time` *é seguida de outra expressão relativa* `/Sigla`. *Este é o caso portanto de utilização da regra* (TRRPE2). *Aplicando-a em (1) acima obtém-se:*

$$\begin{aligned}
&\mathbf{Tr_{APE}}(\texttt{/Time/Sigla}) =\\
&\quad \mathbf{Tr_{REL}}(\texttt{Root}\ \varepsilon\ \texttt{Time})\ \mathbf{Tr_{PRE}}(\texttt{Time}, \varepsilon)\ /\ \mathbf{Tr_{RPE}}(\texttt{Time}, \texttt{Sigla})
\end{aligned} \tag{2}$$

É utilizada a regra (TRREL) *em (2) para traduzir o relacionamento:*

$$\mathbf{Tr_{APE}}(\texttt{/Time/Sigla}) = M_1(\texttt{Root}\ \varepsilon\ \texttt{Time})\ \mathbf{Tr_{PRE}}(\texttt{Time}, \varepsilon)\ /\ \mathbf{Tr_{RPE}}(\texttt{Time}, \texttt{Sigla}) \quad (3)$$

Para traduzir o predicado vazio é utilizada a regra (TRPRE7) *em (3):*

$$\mathbf{Tr_{APE}}(\texttt{/Time/Sigla}) = M_1(\texttt{Root}\ \varepsilon\ \texttt{Time})\ \varepsilon\ /\ \mathbf{Tr_{RPE}}(\texttt{Time}, \varepsilon\ \texttt{Sigla}\ \varepsilon) \quad (4)$$

Nesse momento a regra (TRRPE1) *é utilizada por se tratar de uma expressão relativa sem nenhuma outra expressão seguindo-a.*

$$\mathbf{Tr_{APE}}(\texttt{/Time/Sigla}) = M_1(\texttt{Root}\ \varepsilon\ \texttt{Time})\ /\ \mathbf{Tr_{REL}}(\texttt{Time}\ \varepsilon\ \texttt{Sigla})\ \mathbf{Tr_{PRE}}(\texttt{Sigla}, \varepsilon) \quad (5)$$

Aplicando a regra (TRREL) *em (5), obtém-se:*

$$\mathbf{Tr_{APE}}(\texttt{/Time/Sigla}) = M_1(\texttt{Root}\ \varepsilon\ \texttt{Time})\ /\ M_1(\texttt{Time}\ \varepsilon\ \texttt{Sigla})\ \mathbf{Tr_{PRE}}(\texttt{Sigla}, \varepsilon) \quad (6)$$

Aplicando a regra (TRPRE7) *em (6), é obtido:*

$$\mathbf{Tr_{APE}}(\texttt{/Time/Sigla}) = M_1(\texttt{Root}\ \varepsilon\ \texttt{Time})\ /\ M_1(\texttt{Time}\ \varepsilon\ \texttt{Sigla})\ \varepsilon$$

Finalmente usando as informações de mapeamento dadas na Tabela 3.1 a tradução da consulta CXPath `/Time/Sigla` *para a fonte XML 1 fica:*

$$\mathbf{Tr_{APE}}(\texttt{/Time/Sigla}) = \texttt{/jogadores/jogador/time/sigla}$$

■

Exemplo 3.4 *Este exemplo mostra a tradução para a fonte XML 2 da consulta CXPath* `/Time[Capitão="Dunga"]` *. A consulta busca todas as instâncias do conceito* `Time` *que estão relacionadas com instâncias de* `Capitão` *com valor igual a* `Dunga`*. Primeiramente é utilizada a regra* (TRAPE2) *por se tratar de* $/RelativePath$*:*

$$\mathbf{Tr_{APE}}(\texttt{/Time[Capitão = "Dunga"]}) = \mathbf{Tr_{RPE}}(\texttt{Root}, \varepsilon\ \texttt{Time[Capitão = "Dunga"]}) \quad (1)$$

Para traduzir a expressão relativa é aplicada a regra (TRRPE1) *em (1):*

$$\mathbf{Tr_{APE}}(\texttt{/Time[Capitão = "Dunga"]}) = \mathbf{Tr_{REL}}(\texttt{Root}\ \varepsilon\ \texttt{Time})\ \mathbf{Tr_{PRE}}(\texttt{Time}, \texttt{[Capitão="Dunga"]}) \quad (2)$$

Nesse ponto, é traduzido o relacionamento aplicando a regra (TRREL) *em (2):*

$$\mathbf{Tr_{APE}}(\texttt{/Time[Capitão = "Dunga"]}) = M_2(\texttt{Root}\ \varepsilon\ \texttt{Time})\ \mathbf{Tr_{PRE}}(\texttt{Time}, \texttt{[Capitão="Dunga"]}) \quad (3)$$

Traduzindo o predicado em (3) através da regra (TRPRE6)*:*

$$\mathbf{Tr_{APE}}(\texttt{/Time[Capitão = "Dunga"]}) = M_2(\texttt{Root}\ \varepsilon\ \texttt{Time})\ \texttt{[}\ \mathbf{Tr_{RPE}}(\texttt{Time}, \varepsilon\ \texttt{Capitão}\ \varepsilon)\texttt{="Dunga"}\ \mathbf{Tr_{PRE}}(\texttt{Time}, \varepsilon) \quad (4)$$

Aplicando a regra (TRRPE1) *em (4), obtém-se:*

$$\begin{aligned}&\mathbf{Tr_{APE}}(\texttt{/Time[Capitão = "Dunga"]})\\&\quad M_2(\texttt{Root}\ \varepsilon\ \texttt{Time})\ [\ \mathbf{Tr_{REL}}(\texttt{Time}\ \varepsilon\ \texttt{Capitão})\ \mathbf{Tr_{PRE}}(\texttt{Capitão}, \varepsilon)\\&\quad =\texttt{"Dunga"]}\,\mathbf{Tr_{PRE}}(\texttt{Time}, \varepsilon)\end{aligned} \tag{5}$$

Aplicando (TRREL) *em (5):*

$$\begin{aligned}&\mathbf{Tr_{APE}}(\texttt{/Time[Capitão = "Dunga"]}) =\\&\quad M_2(\texttt{Root}\ \varepsilon\ \texttt{Time})\ [\ M_2(\texttt{Time}\ \varepsilon\ \texttt{Capitão})\\&\quad \mathbf{Tr_{PRE}}(\texttt{Capitão}, \varepsilon) =\texttt{"Dunga"]}\ \mathbf{Tr_{PRE}}(\texttt{Time}, \varepsilon)\end{aligned} \tag{6}$$

Utilizando a regra (TRPRE7) *em (6):*

$$\begin{aligned}&\mathbf{Tr_{APE}}(\texttt{/Time[Capitão = "Dunga"]}) =\\&\quad M_2(\texttt{Root}\ \varepsilon\ \texttt{Time})\ [\ M_2(\texttt{Time}\ \varepsilon\ \texttt{Capitão})\ \varepsilon\texttt{="Dunga"]}\ \varepsilon\end{aligned} \tag{7}$$

Obtém-se de (7) *a tradução da consulta* `/Time[Capitão="Dunga"]`*:*

$$\begin{aligned}&\mathbf{Tr_{APE}}(\texttt{/Time[Capitão = "Dunga"]}) =\\&\qquad\qquad \texttt{/times/time[capitão="Dunga"]}\end{aligned}$$

■

3.4 Tradução Levando em Consideração Herança

Na Figura 3.4 é mostrado um modelo conceitual que possui um relacionamento de herança, criado a partir do esquema XML da Figura 3.5 . O relacionamento de herança ocorre entre os conceitos `Jogador` (conceito genérico) e `JogadorReserva` (conceito especializado), sendo assim, o conceito `JogadorReserva` herdará todos os relacionamentos do conceito `Jogador` com outros conceitos do modelo conceitual. Na Tabela 3.3 é mostrado o conjunto de informações de mapeamento para a fonte representada pelo esquema da Figura 3.5. As informações em negrito na tabela representam as informações do subconceito `JogadorReserva` e seus relacionamentos herdados de `Jogador`.

Na Tabela 3.3 a informação de mapeamento do elemento `Time` ε `JogadorReserva`, por exemplo, possui a consulta *XPath* `..[local-name(.)="reserva"]`. Isso significa que, é necessário subir um nível na árvore XML, porém, apenas se o elemento XML que representa o conceito `Time` for filho do elemento XML `reserva`. É necessário essa consulta *XPath* pois, para o elemento do modelo conceitual `Time` é especificada a consulta `/jogadores/*/time`, o que abrange os elementos XML `jogador` e `reserva`. Então para fazer a distinção entre esses elementos é utilizada a expressão `local-name(.)`.

Com as informações dos relacionamentos de herança refletidas nas informações de mapeamento, o mecanismo de tradução permanece intacto, ou seja, não há necessidade de mudança do mecanismo de tradução proposto afim de lidar com esse tipo de relacionamento.

3.5 Mecanismo de Tradução $\times$ Expressividade do Modelo Conceitual

Quando realizada a consulta *CXPath* `/Jogador/Nome` sobre o modelo conceitual da Figura 3.2, perguntando pelo nome de todos os jogadores, uma situação interessante

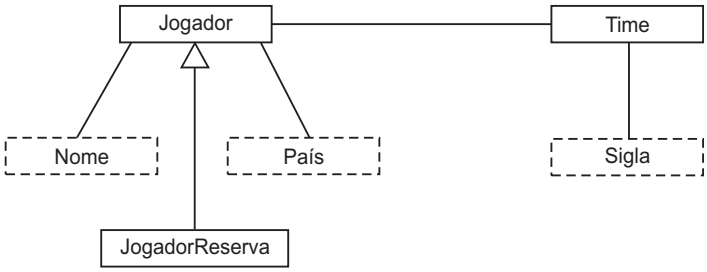


Figura 3.4: Modelo Conceitual com herança.

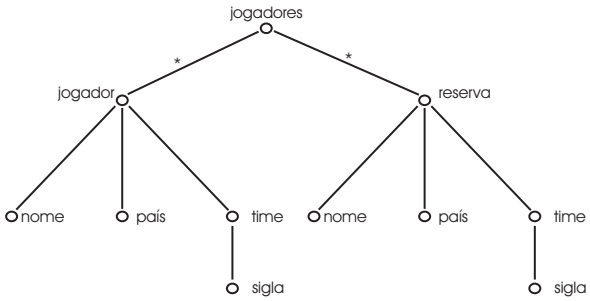


Figura 3.5: Esquema XML.

Tabela 3.3: Informações de Mapeamento com herança.

Conceito/Relacionamento	Fonte XML 1
Root ε Jogador	`/jogadores/*`
Jogador ε Time	`time`
Jogador ε Nome	`nome`
Jogador ε País	`pais`
Time ε Jogador	..
Nome ε Jogador	..
País ε Jogador	..
Root ε JogadorReserva	`/jogadores/reserva`
JogadorReserva ε Time	`time`
JogadorReserva ε Nome	`nome`
JogadorReserva ε País	`pais`
Time ε JogadorReserva	`..[local-name(.)="reserva"]`
Nome ε JogadorReserva	`..[local-name(.)="reserva"]`
País ε JogadorReserva	`..[local-name(.)="reserva"]`
Root ε Time	`/jogadores/*/time`
Root ε Sigla	`/jogadores/*/time/sigla`
Time ε Sigla	`sigla`
Sigla ε Time	..

ocorrerá. Na fonte XML representada pelo esquema da Figura 3.3(a), a consulta *CXPath* `/Jogador/Nome` seria traduzida para a consulta `/jogadores/jogador/nome` sobre a fonte. Entretanto, na fonte XML cujo esquema é apresentado na Figura 3.3(b), não seria possível uma tradução. Observando o modelo conceitual e utilizando nosso conhec-

imento sobre o domínio, podemos dizer que todo capitão de time também é um jogador. Mais especificamente, cada capitão de um time, nesse modelo conceitual, representa o nome de um jogador.

Afim de melhorar o resultado da consulta *CXPath* `/Jogador/Nome` além de pesquisar o nome dos jogadores, nesse caso, seriam também pesquisados os capitães de times. Com a tradução da consulta utilizando o mecanismo proposto nesse capítulo esses resultados não seriam contemplados.

Com o intuito de obter melhores resultados de consultas, foi aumentada a expressividade do modelo conceitual, permitindo a especificação de restrições de integridade sob o esquema global. Com o aumento da expressividade do modelo conceitual é possível inferir resultados que antes não seriam contemplados com a simples tradução de uma consulta.

No próximo capítulo são apresentadas as restrições de integridade utilizadas sob o modelo conceitual e a implicação delas na resolução de consultas dos sistemas de integração de dados deste trabalho.

4 RESOLUÇÃO DE CONSULTAS UTILIZANDO DEPENDÊNCIAS DE INCLUSÃO

Existem situações em que a simples tradução de uma consulta *CXPath* não contempla alguns resultados, pois as fontes de dados podem ser incompletas. Neste capítulo, o modelo conceitual, que constitui o esquema global do sistema de integração de dados, é estendido com dependências de inclusão e o mecanismo de resolução de consultas é modificado para lidar com esse tipo de dependência.

Com o aumento de expressividade do esquema global é possível inferir resultados, a partir dos dados disponíveis no sistema de integração, que antes não seriam contemplados com a simples tradução de uma consulta.

4.1 O Papel das Restrições de Integridade

Em integração de dados, o esquema global é utilizado com o intuito de abstrair as fontes envolvidas no processo de integração, isentando o usuário do conhecimento da estrutura de cada fonte e de onde estão os dados de interesse. Como o esquema global atua como uma interface pela qual os usuários submetem consultas ao sistema de integração, tal esquema deve incorporar mecanismos flexíveis de representação para relacionar os vários elementos do esquema global de acordo com a semântica do domínio de interesse. As restrições de integridade fazem exatamente esse papel declarando as inter-relações entre os elementos de um esquema (ABITEBOUL; HULL; VIANU, 1995).

As restrições de integridade são declarações que indicam o significado pretendido dos dados armazenados em um banco de dados. Elas são tradicionalmente parte da especificação do esquema, pois complementam a descrição da estrutura dos dados. As restrições de integridade expressam propriedades que devem ser satisfeitas por todas as instâncias de um esquema de banco de dados.

Neste trabalho é feita a especificação de dependências de inclusão sobre o modelo conceitual (esquema global) afim de aumentar a sua expressividade. Em sistemas de integração de dados geralmente as fontes são autônomas, ou seja, os dados extraídos podem não satisfazer as restrições do esquema global. A informação especificada nas dependências de inclusão sobre o esquema global, neste trabalho, permite obter respostas adicionais que não seriam providas pela simples tradução de uma consulta.

4.2 Dependência de Inclusão

Considere o esquema global da Figura 4.1. De forma a expressar que todo valor do conceito léxico `Capitão`, relacionado com o conceito não-léxico `Time`, também é um

valor do conceito `Nome` relacionado com o conceito `Jogador`, é especificada a seguinte dependência de inclusão:

$$\texttt{Time.Capitão} \subseteq \texttt{Jogador.Nome}$$

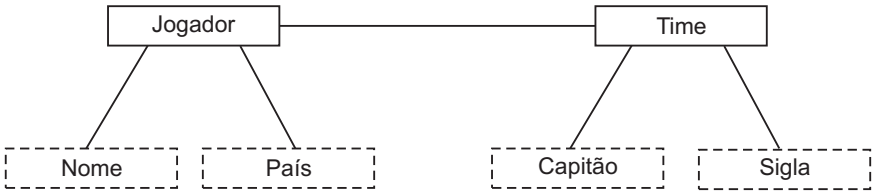


Figura 4.1: Modelo Conceitual.

No exemplo acima, o conceito não léxico `Time` está em um relacionamento sem nome com o conceito léxico `Capitão` (idem para o conceito não-léxico `Jogador` e o seu relacionamento com o conceito léxico `Nome`). O caso mais geral permite a especificação de dependências de inclusão envolvendo relacionamentos com nomes:

$$nl_1.Rel_1\ l_1 \ \subseteq \ nl_2.Rel_2\ l_2,$$

onde Rel é o nome do relacionamento.

Dependências de inclusão, nessa forma, expressam que os valores de um dado conceito léxico l_1, relacionado com um conceito não-léxico nl_1 através de um relacionamento de nome Rel_1, também são valores de um conceito léxico l_2 relacionado com o conceito não-léxico nl_2 através do relacionamento de nome Rel_2.

Uma dependência de inclusão $nl_1.Rel_1\ l_1 \subseteq nl_2.Rel_2\ l_2$ é dita *bem formada* em relação a um modelo conceitual $\mathcal{CM} = \langle \mathcal{NL}, \mathcal{L}, \mathcal{R} \rangle$ se nl_1, nl_2 e l_1, l_2 são conceitos não-léxicos e léxicos, respectivamente, pertencentes ao modelo conceitual $\mathcal{CM}$, ou seja:

i) $nl_1, nl_2 \in \mathcal{NL}$ e $l_1, l_2 \in \mathcal{L}$,

e se existir, no modelo conceitual, relacionamento entre os conceitos nl_1 e l_1 de nome Rel_1 e relacionamento entre os conceitos nl_2 e l_2 de nome Rel_2, ou seja:

ii) existe $r \in \mathcal{R}$ tal que $r.id_1 = nl_1,\ r.id_2 = l_1,\ r.n = Rel_1$ e
existe $r' \in \mathcal{R}$ tal que $r'.id_1 = nl_2,\ r'.id_2 = l_2, r'.n = Rel_2$.

Dado um banco de dados $\mathcal{B}$ para um modelo conceitual $\mathcal{CM}$, $\mathcal{B}$ respeita uma dependência de inclusão, escrito na forma $\mathcal{B} \models nl_1.Rel_1\ l_1 \subseteq nl_2.Rel_2\ l_2$, se e somente se:

- para todo valor do conceito léxico l_1 que está em um relacionamento Rel_1 com o conceito não-léxico nl_1, existe um conceito léxico l_2, em um relacionamento Rel_2 com o conceito não-léxico nl_2, com o mesmo valor semântico de l_1.

Dado um banco de dados $\mathcal{B}$ para um modelo conceitual $\mathcal{CM}$, se $\mathcal{B}$ respeita as dependências de inclusão $nl_1.Rel_1\ l_1 \subseteq nl_2.Rel_2\ l_2$ presentes em um conjunto $\mathcal{ID}$, é dito que $\mathcal{B}$ é consistente em relação à $\mathcal{ID}$, denotado $\mathcal{B} \models \mathcal{ID}$.

4.3 Reescrita de Consultas

Uma opção para resolver consultas em um sistema de integração de dados é submeter a consulta sobre uma base global contendo apenas os dados retornados quando os mapeamentos são aplicados sobre as fontes. Porém, tal base pode não respeitar as restrições impostas pelo esquema, pois as fontes podem ser incompletas. Nesse caso, seria necessário raciocinar sobre as restrições com o intuito de tornar essa base global consistente.

O *chase* (ABITEBOUL; HULL; VIANU, 1995) é um mecanismo que aplica exaustivamente um conjunto de regras para transformar um banco de dados a fim de torná-lo consistente de acordo com um conjunto de restrições de integridade. Existe apenas uma regra do *chase* que trata as dependências de inclusão: um novo fato (tupla) é adicionado ao banco de dados toda vez que uma dependência de inclusão não é satisfeita.

A aplicação da *chase* pode ser um processo longo no caso de um banco de dados com várias restrições de integridade e um grande número de instâncias. Pode existir também uma aplicação infinita das regras do *chase*, como é o caso na existência de dependências de inclusão cíclicas. Por esses motivos, na maioria das vezes, a utilização do *chase* não é a melhor alternativa.

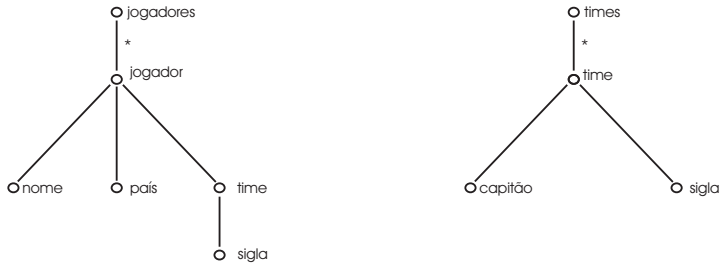
Ao invés de aplicar o *chase* no banco de dados (instâncias), uma idéia é aplicar o seu princípio às consultas, aumentando consideravelmente a eficiência do sistema, pois a maior parte do processamento de consultas ficará no nível intensional (CALÌ et al., 2004). Com isso, cada consulta será reescrita de forma que sejam embutidas as informações sobre as restrições. Um algoritmo que segue esse princípio, no contexto de *conjunctive queries*, é o IDrewrite (CALÌ et al., 2003).

O desenvolvimento do mecanismo de reescrita proposto neste capítulo segue essa mesma filosofia, ou seja, novas consultas *CXPath* são produzidas, nas quais as informações das dependências de inclusão estarão codificadas, a partir das consultas *CXPath* originais. A consulta *CXPath* original e as resultantes do processo de reescrita são traduzidas para consultas *XPath* sobre as fontes pelo mecanismo de tradução proposto no Capítulo 3.

Exemplo 4.1 *Seja $\mathcal{I} = \langle \mathcal{G}, \mathcal{S}, \mathcal{M} \rangle$ um sistema de integração de dados, onde $\mathcal{G}$ é modelo conceitual da Figura 4.1, estendido pela seguinte dependência de inclusão:*

$$\texttt{Time.Capitão} \subseteq \texttt{Jogador.Nome}$$

O conjunto $\mathcal{S}$ é constituído pelos esquemas das fontes XML apresentados na Figura 4.2, e os mapeamentos $\mathcal{M}$ entre o esquema global e as fontes são mostrados na Tabela 4.1.



(a) Esquema XML 1 (b) Esquema XML 2

Figura 4.2: Esquemas das fontes XML.

Tabela 4.1: Informações de Mapeamento dos esquemas XML da Figura 4.2.

Conceito/Relacionamento	Fonte XML 1	Fonte XML 2
Root ε Jogador	`/jogadores/jogador`	`indefinido`
Root ε Time	`/jogadores/jogador/time`	`/times/time`
Root ε País	`/jogadores/jogador/país`	`indefinido`
Root ε Nome	`/jogadores/jogador/nome`	`/times/time/capitão`
Root ε Sigla	`/jogadores/jogador/time/sigla`	`/times/time/sigla`
Root ε Capitão	`indefinido`	`/times/time/capitão`
Jogador ε País	`país`	`indefinido`
Jogador ε Nome	`nome`	`indefinido`
Jogador ε Time	`time`	`indefinido`
Time ε Jogador	..	`indefinido`
Time ε Sigla	`sigla`	`sigla`
Time ε Capitão	`indefinido`	`capitão`
País ε Jogador	..	`indefinido`
Nome ε Jogador	..	`indefinido`
Sigla ε Time	..	..
Capitão ε Time	`indefinido`	..

Considere a consulta CXPath `/Jogador/Nome`, para recuperar o nome de todos os jogadores, e as fontes XML da Figura 4.3. Fazendo a tradução da consulta CXPath sem levar em consideração a dependência de inclusão, o resultado da consulta seria `{Léo Moura, Ibson, Nilmar}`. Porém, por causa da dependência de inclusão é de conhecimento que `Fábio Luciano` e `Fernandão` são também nomes de jogadores. Utilizando o mecanismo a ser definido na Seção 4.4 a consulta original `/Jogador/Nome` é reescrita para a consulta CXPath `/Time/Capitão` de forma a obter os nomes de jogadores não contemplados pela consulta original.

■

4.4 Mecanismo de Reescrita de Consultas Utilizando Dependências de Inclusão

Nesta seção é apresentado o mecanismo de reescrita de consultas afim de incorporar nas consultas globais as informações das dependências de inclusão especificadas sobre o esquema global do sistema de integração de dados. Dada uma consulta *CXPath* e uma dependência de inclusão, o mecanismo de reescrita Rew gera uma nova consulta *CXPath*.

$$\mathsf{Rew} : \mathsf{CXPath} \times \mathcal{ID} \rightarrow \mathsf{CXPath}$$

O mecanismo de reescrita Rew é constituído pelas funções $\mathbf{Rew_{APE}}$, $\mathbf{Rew_{RPE}}$, $\mathbf{Rew_{REL}}$ e $\mathbf{Rew_{PRE}}$. Elas tratam, respectivamente, expressões de caminho absoluto, expressões de caminho relativo, relacionamentos, e predicados.

Por questão de legibilidade, a informação correspondente à dependência de inclusão é omitida nas regras de reescrita especificadas. Essa informação aparece explicitamente na função de reescrita $\mathbf{Rew_{REL}}$ (Subseção 4.4.3) onde é feita a verificação da inclusão.

A idéia essencial por trás da reescrita é a da *substituição*: uma consulta *CXPath* é percorrida e, toda vez que um relacionamento entre conceitos é encontrado, é verificado

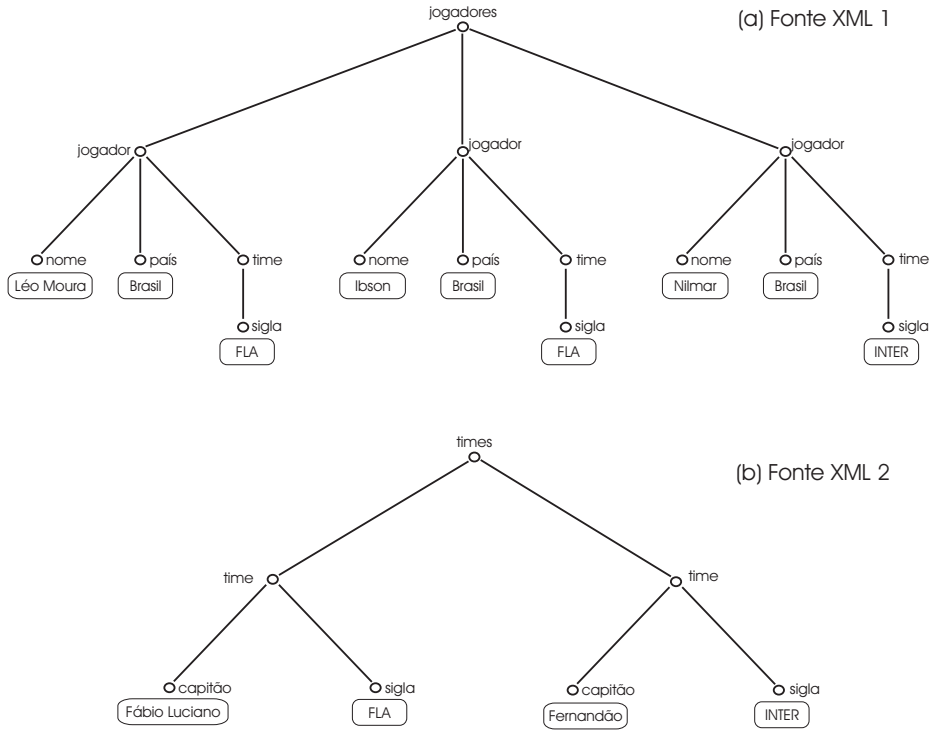


Figura 4.3: Fontes XML.

se a dependência de inclusão, passada como argumento, está associada a esse relacionamento. Se estiver, uma substituição será produzida. A geração da nova consulta é feita pela propagação das substituições produzidas a toda consulta. A origem dessa substituição e a sua correta propagação são cruciais a esse processo de reescrita.

As substituições são representadas através das metavariáveis σ, σ' e especificadas por $[id_1 \mapsto id_2]$, que substitui um conceito id_1 por um conceito id_2 quando aplicadas a uma consulta. Por exemplo, se uma substituição $\sigma =$ `[Jogador` $\mapsto$ `Time]` é aplicada sobre a consulta `/Jogador/Capitão`, o resultado seria:

$$\begin{aligned}\sigma\texttt{(/Jogador/Capitão)} &= \texttt{[Jogador} \mapsto \texttt{Time](/Jogador/Capitão)}\\ &= \texttt{/Time/Capitão}\end{aligned}$$

4.4.1 Reescrita de Expressões de Caminho Absoluto ($\mathbf{Rew_{APE}}$)

Expressões de caminho absoluto são reescritas pela função $\mathbf{Rew_{APE}}$. Pela regra (REWAPE1) a reescrita da expressão de caminho absoluto "/" produz a própria "/" na saída.

(REWAPE1) $\mathbf{Rew_{APE}}(/) = /$

Pela regra (REWAPE2) uma expressão de caminho absoluto $/RelativePath$ é reescrita para $/relpath$, onde $relpath$ é obtido aplicando a função $\mathbf{Rew_{RPE}}$ (Subseção 4.4.2) à $RelativePath$, passando como contexto o conceito $Root$.

$$\begin{aligned}\text{(REWAPE2)}\ \mathbf{Rew_{APE}}\ &(/RelativePath) =\\ &let\ (\sigma, relpath) :=\ \mathbf{Rew_{RPE}}(Root, RelativePath)\\ &in\ \ / relpath\end{aligned}$$

4.4.2 Reescrita de Expressões de Caminho Relativo ($\mathbf{Rew_{RPE}}$)

Expressões de caminho relativo são reescritas pela função $\mathbf{Rew_{RPE}}$. A regra (REWRPE1) primeiramente reescreve o predicado $Preds$ passando como contexto id_2, utilizando a função $\mathbf{Rew_{PRE}}$ que trata da reescrita dos predicados (Subseção 4.4.4).

$$
\begin{aligned}
\text{(REWRPE1)}\ \mathbf{Rew_{RPE}}\ &(id_1, Rel\ id_2\ Preds) = \\
&let\ (\sigma, preds) \ := \ \mathbf{Rew_{PRE}}(id_2, Preds), \\
&\quad (\sigma', rel\ id_2') := \ \mathbf{Rew_{REL}}(id_1,\ Rel\ id_2) \\
&in\ \ (\sigma' \circ \sigma,\ \sigma(rel\ id_2')\ preds)
\end{aligned}
$$

Essa reescrita gera um par $(\sigma, preds)$, onde σ é uma substituição, e $preds$ é o predicado resultante. É aplicada também a função $\mathbf{Rew_{REL}}$ para verificar a existência da informação da dependência de inclusão nos argumentos id_1 e $Rel\ id_2$. Essa função retorna uma substituição σ' e o resultado $rel\ id_2'$.

A regra (REWRPE1) tem como resultado final um par: é aplicada à $rel\ id_2'$ a substituição σ e esse resultado é concatenado a $preds$; o outro componente é a substituição a ser realizada pela composição das substituições σ e σ'. A composição das substituições é importante pois, uma substituição pode ser gerada tanto em σ quanto σ'.

A regra (REWRPE2) aplica a função $\mathbf{Rew_{RPE}}$ às diferentes expressões de caminho relativo, obtendo em cada aplicação dessa regra uma substituição (σ e σ') e um resultado ($rel\ id_2'\ preds$ e $relpath$). O resultado da aplicação da regra (REWRPE2) consiste em uma substituição, que é a composição de σ e σ', e o resultado da aplicação de σ à $rel\ id_2'\ preds$ concatenado à $/\ relpath$.

$$
\begin{aligned}
\text{(REWRPE2)}\ \mathbf{Rew_{RPE}}\ &(id_1, Rel\ id_2\ Preds/RelPath) = \\
&let\ (\sigma, rel\ id_2'\ preds) \ := \ \mathbf{Rew_{RPE}}(id_1, Rel\ id_2\ Preds), \\
&\quad (\sigma', relpath) \qquad := \ \mathbf{Rew_{RPE}}(id_2, RelPath) \\
&in\ \ (\sigma' \circ\ \sigma,\ \sigma'(rel\ id_2'\ preds)/\ relpath)
\end{aligned}
$$

4.4.3 Relacionamentos ($\mathbf{Rew_{REL}}$)

A função $\mathbf{Rew_{REL}}$ faz a verificação da existência das dependências de inclusão na parte da consulta que está sendo analisada. A regra (REWREL) especifica que, se os argumentos id_1 e $Rel\ id_2$ correspondem à informação $nl_2.Rel_2\ l_2$ da dependência de inclusão passada como parâmetro, a informação $Rel\ id_2$ é reescrita para $Rel_1\ l_1$, especificada na dependência de inclusão passada como argumento.

Além disso, é gerada uma substituição do conceito id_1 pelo conceito nl_1, afim de refletir a informação sobre a dependência de inclusão na consulta analisada. Se não for encontrada a informação da dependência de inclusão, a regra retorna uma substituição vazia, e o relacionamento Rel juntamente com o conceito id_2, ou seja, não é realizada nenhuma reescrita na parte da consulta verificada.

$$
\begin{aligned}
\text{(REWREL)}\ \mathbf{Rew_{REL}}\ &(id_1,\ \ Rel\ id_2,\ \ nl_1.Rel_1\ l_1\ \subseteq\ nl_2.Rel_2\ l_2) = \\
&se\ (id_1 = nl_2\ \wedge\ id_2 = l_2\ \wedge\ Rel = Rel_2)\ então \\
&\qquad ([\ id_1 \mapsto nl_1\ ], Rel_1\ l_1) \\
&senão \\
&\qquad ([\ ], Rel\ id_2)
\end{aligned}
$$

Pode-se notar que essa é a única regra que constrói, de fato, uma substituição, as demais regras propagam e compõem substituições geradas.

4.4.4 Reescrita dos Predicados ($\mathbf{Rew_{PRE}}$)

As regras que avaliam os predicados (REWPRE1) e (REWPRE2) aplicam as regras de expressões de caminho absolutas e relativas, respectivamente, aos próprios predicados, conservando os colchetes **[ ]**, os operadores *op* e os literais $Literal$. É aplicada também recursivamente a regra de reescrita no caso da existência de mais de um predicado. A regra (REWPRE3) é utilizada quando não existem predicados e retorna uma substituição vazia e um resultado também vazio.

$$\begin{aligned}(\text{REWPRE1})\ \mathbf{Rew_{PRE}}\ &(id, [/RelPath\ op\ Literal]\ Preds) = \\ &let\ (\sigma, relpath) := \mathbf{Rew_{RPE}}(Root, RelPath), \\ &\quad (\sigma', preds) := \mathbf{Rew_{PRE}}(id, Preds) \\ &in\ \ (\sigma' \circ \sigma, [\ /relpath\ op\ Literal\ ]\ preds)\end{aligned}$$

$$\begin{aligned}(\text{REWPRE2})\ \mathbf{Rew_{PRE}}&(id, [RelPath\ op\ Literal]\ Preds) = \\ &let\ (\sigma, relpath) := \mathbf{Rew_{RPE}}(id, RelPath), \\ &\quad (\sigma', preds) := \mathbf{Rew_{PRE}}(id, Preds) \\ &in\ \ (\sigma' \circ \sigma, [\ relpath\ op\ Literal\ ]\ preds)\end{aligned}$$

$$(\text{REWPRE3})\ \mathbf{Rew_{PRE}}(id, \varepsilon) = [\,],\ \varepsilon$$

Não foram consideradas todas as possibilidades sintáticas da linguagem *CXPath* para predicados. As regras que faltam seguem diretamente das regras já propostas.

Utilizando o mecanismo proposto acima, a consulta `/Jogador/Nome`, através da dependência de inclusão `Time.Capitão` $\subseteq$ `Jogador.Nome` seria reescrita para `/Time/Capitão`. A consulta original é percorrida e quando encontrado o relacionamento entre os conceitos `Jogador` e `Nome`, primeiramente ocorre a troca do conceito `Nome` pelo conceito `Capitão` (`/Jogador/Capitão`) e geração da substituição `[Jogador` $\mapsto$ `Time]`, e depois ocorre a propagação da substituição gerada para que o conceito `Jogador` seja trocado para o conceito `Time` (`/Time/Capitão`).

Para maiores detalhes sobre o mecanismo de reescrita observar o Exemplo 4.2 que apresenta passo-a-passo a aplicação das regras desenvolvidas no mecanismo de reescrita proposto. O exemplo abaixo leva em consideração a dependência de inclusão $\mathcal{ID} = \{$ `Time.Capitão` $\subseteq$ `Jogador.Nome`$\}$ para realização da reescrita.

Exemplo 4.2 *Considere a consulta CXPath de entrada* `/Jogador/Nome`*. Para expressões de caminho absoluto existem duas regras: (*REWAPE1*) quando se trata apenas de uma "/" e (*REWAPE2*) quando se trata de uma "/" seguida de uma expressão relativa. Nesse caso será utilizada a regra (*REWAPE2*) para reescrever a consulta.*

$$\begin{aligned}(I)\ \ &Rew_{APE}\ (\texttt{/Jogador/Nome}) = \\ &\quad let\ (\sigma, relpath) := \underline{Rew_{RPE}(\texttt{Root, Jogador/Nome})^{II}}\ in\ /relpath\end{aligned}$$

Pode-se notar que para conseguir o resultado da expressão (I) *é necessário obter o resultado da expressão* (II)*, que é avaliada utilizando a função* $\mathbf{Rew_{RPE}}$*. Mais especificamente é utilizada a regra (*REWRPE2*), pois se trata da expressão relativa* (ε `Jogador` ε) *seguida da expressão relativa* `(Nome)`*.*

$$\begin{aligned}(II)\ \ &Rew_{RPE}\ (\texttt{Root}, \varepsilon\ \texttt{Jogador}\ \varepsilon\texttt{/Nome}) = \\ &\quad let\ \ (\sigma, rel\ id_2'\ preds) := \overline{Rew_{RPE}(\texttt{Root}, \varepsilon\texttt{Jogador}\ \varepsilon)^{III}}, \\ &\qquad (\sigma', relpath) \quad := \overline{Rew_{RPE}(\texttt{Jogador, Nome})^{IV}} \\ &\quad in\ \ (\sigma' \circ \sigma,\ \sigma'(rel\ id_2'\ preds)/\ relpath)\end{aligned}$$

Mais uma vez, para obtenção do resultado da expressão (II) *é necessário a avaliação das expressões* (III) *e* (IV)*, ambas avaliadas pela função* $\mathbf{Rew_{RPE}}$*. Primeiramente será avaliada a expressão* (III) *até ela ser totalmente avaliada, para depois ser avaliada a expressão* (IV)*. A expressão* (III) *é avaliada através da regra (*REWRPE1*) por se tratar apenas da expressão relativa* $(\varepsilon\ \texttt{Jogador}\ \varepsilon)$*.*

$$
\begin{aligned}
(III)\ & Rew_{RPE}(\texttt{Root}, \varepsilon\ \texttt{Jogador}\ \varepsilon) = \\
& \quad let\ (\sigma, preds)\ := \underline{Rew_{PRE}(\texttt{Jogador}, \varepsilon)}^{V} \\
& \qquad (\sigma', rel\ id'_2) := \underline{Rew_{REL}(\texttt{Root},\ \varepsilon\ \texttt{Jogador}, \texttt{Time.Capitão} \subseteq \texttt{Jogador.Nome})}^{VI} \\
& \quad in\ (\sigma' \circ \sigma,\ \sigma(rel\ id'_2)\ preds)
\end{aligned}
$$

O resultado da expressão (III) *é obtido através da avaliação das expressões* (V) *e* (VI)*. A avaliação da expressão* (V) *é realizada através da regra (*REWRPE3*) pois se trata de um predicado vazio. No resultado é gerado uma substituição vazia e um elemento vazio que corresponde ao predicado vazio:*

$$(V)\ \ Rew_{PRE}(\texttt{Jogador}, \varepsilon) = [\,],\ \varepsilon$$

Um dos resultados da expressão (III) *foi obtido, basta agora computar o resultado da expressão* (VI) *através da função* $\mathbf{Rew_{REL}}$*.*

$$
\begin{aligned}
(III)\ & Rew_{RPE}(\texttt{Root}, \varepsilon\ \texttt{Jogador}\ \varepsilon) = \\
& \quad let\ (\sigma, preds)\ := ([\,],\ \varepsilon) \\
& \qquad (\sigma', rel\ id'_2) := \underline{Rew_{REL}(\texttt{Root},\ \varepsilon\ \texttt{Jogador},\ \texttt{Time.Capitão} \subseteq \texttt{Jogador.Nome})}^{VI} \\
& \quad in\ (\sigma' \circ [\,],\ [\,](rel\ id'_2)\ \varepsilon)
\end{aligned}
$$

Na avaliação da expressão (VI) *utilizando a regra (*REWREL*) é verificada se a informação da dependência de inclusão está presente na parte da consulta CXPath avaliada. Nesse caso, o resultado é uma substituição vazia juntamente com o o relacionamento* Rel *e o conceito identificado como* id_2*.*

$$(VI)\ \ Rew_{REL}(\texttt{Root},\ \varepsilon\ \texttt{Jogador}) = [\,],\ \varepsilon\ \texttt{Jogador}$$

Com isso é obtido o resultado da expressão (III)*:*

$$
\begin{aligned}
(III)\ \ & Rew_{RPE}\ (\texttt{Root}, \varepsilon\ \texttt{Jogador}\ \varepsilon) = \\
& \quad let\ \ (\sigma, preds)\ \ := \ ([\,],\ \varepsilon) \\
& \qquad (\sigma', rel\ id'_2) := \ ([\,],\ \texttt{Jogador}) \\
& \quad in\ \ \underline{([\,] \circ [\,], [\,](\texttt{Jogador})\ \varepsilon)} \equiv \underline{(\ [\,], \texttt{Jogador})}
\end{aligned}
$$

Nesse momento, obtido o resultado da expressão (III)*, afim de resolver a expressão* (II)*, é avaliada a expressão* (IV)*.*

$$
\begin{aligned}
(II)\ \ & Rew_{RPE}\ (\texttt{Root}, \varepsilon\ \texttt{Jogador}\ \varepsilon\texttt{/Nome}) = \\
& \quad let\ \ (\sigma, rel\ id'_2\ preds) := \ (\ [\,], \texttt{Jogador}), \\
& \qquad (\sigma', relpath) \qquad := \underline{Rew_{RPE}(\texttt{Jogador,Nome})}^{IV} \\
& \quad in\ \ (\sigma' \circ [\,],\ \sigma'(Jogador)/\ relpath)
\end{aligned}
$$

Para resolver a expressão (IV) *é utilizada a regra (*REWRPE1*) por se tratar apenas da expressão* $(\varepsilon\ \texttt{Nome}\ \varepsilon)$*, com o conceito* $\texttt{Jogador}$ *como contexto. Porém, para obtenção do resultado é necessário a avaliação das expressões* (VII) *e* $(VIII)$ *resolvidas através das funções* $\mathbf{Rew_{PRE}}$ *e* $\mathbf{Rew_{REL}}$ *respectivamente.*

$(IV)\ Rew_{RPE}(\texttt{Jogador}, \varepsilon\ \texttt{Nome}\ \varepsilon) =$
$\quad let\ (\sigma, preds)\ := \underline{Rew_{PRE}(\texttt{Nome}, \varepsilon)}^{VII}$
$\quad\quad (\sigma', rel\ id'_2) := \underline{Rew_{REL}(\texttt{Jogador},\ \varepsilon\ \texttt{Nome},\ \texttt{Time.Capitão} \subseteq \texttt{Jogador.Nome})}^{VIII}$
$\quad in\ (\sigma' \circ \sigma,\ \sigma(rel\ id'_2)\ preds)$

*A expressão (VII) é avaliada pela regra (*REWPRE3*), pois possui um predicado vazio para o conceito* `Nome`*. Com isso:*

$(VII)\ \ Rew_{PRE}(\texttt{Nome}, \varepsilon) = [\,], \varepsilon$

A expressão (IV) necessita agora do resultado da expressão $(VIII)$.

$(IV)\ Rew_{RPE}\texttt{Jogador}, \varepsilon\ Nome\ \varepsilon) =$
$\quad let\ (\sigma, preds)\ := ([\,], \varepsilon)$
$\quad\quad (\sigma', rel\ id'_2) := \underline{Rew_{REL}(\texttt{Jogador},\ \varepsilon\ \texttt{Nome},\ \texttt{Time.Capitão} \subseteq \texttt{Jogador.Nome})}^{VIII}$
$\quad in\ (\sigma' \circ [\,],\ [\,](rel\ id'_2)\ \varepsilon)$

Utilizando a regra (REWREL) *na expressão $(VIII)$ é verificada a informação especificada na dependência de inclusão na porção da consulta* CXPath *analisada. Sendo assim, o conceito* `Nome` *é reescrito para o conceito* `Capitão`*, e é gerada uma substituição para que o conceito* `Jogador` *seja substituído pelo conceito* `Time`*, afim de tornar coerente a reescrita.*

$(VIII)\ \ Rew_{REL}(\texttt{Jogador},\ \varepsilon\ \texttt{Nome},\ \texttt{Time.Capitão} \subseteq \texttt{Jogador.Nome}) =$
$\quad\quad [\texttt{Jogador} \mapsto \texttt{Time}],\ \texttt{Capitão}$

Com o resultado da expressão $(VIII)$ é obtido o resultado da expressão (IV):

$(IV)\ \ Rew_{RPE}(\texttt{Jogador}, \varepsilon\ \texttt{Nome}\ \varepsilon) =$
$\quad let\ (\sigma, preds)\ :=\ ([\,], \varepsilon)$
$\quad\quad (\sigma', rel\ id'_2) := \ ([\texttt{Jogador} \mapsto \texttt{Time}], \texttt{Capitão})$
$\quad in\ \underline{([\texttt{Jogador} \mapsto \texttt{Time}] \circ [\,],\ [\,](\texttt{Capitão})\ \varepsilon)}$
$\quad \equiv\ \underline{([\texttt{Jogador} \mapsto \texttt{Time}], \texttt{Capitão})}$

Agora pode-se obter o resultado da expressão (II).

$(II)\ \ Rew_{RPE}\ (\texttt{Root}, \varepsilon\ \texttt{Jogador}\ \varepsilon\texttt{/Nome}) =$
$\quad\quad let\ (\sigma, rel\ id'_2\ preds)\ :=\ (\ [\,], \texttt{Jogador}),$
$\quad\quad\quad (\sigma', relpath)\ \ :=\ ([\texttt{Jogador} \mapsto \texttt{Time}], \texttt{Capitão})$
$\quad\quad in\ \underline{([\texttt{Jogador} \mapsto \texttt{Time}] \circ [\,],\ [\texttt{Jogador} \mapsto \texttt{Time}](\texttt{Jogador})\texttt{/Capitão})}$
$\quad\quad \equiv\ \underline{([\texttt{Jogador} \mapsto \texttt{Time}], \texttt{Time/Capitão})}$

Por fim o resultado da expressão (I) é obtido:

$(I)\ \ Rew_{APE}\ (\texttt{/Jogador/Nome}) =$
$\quad\quad let\ (\sigma, relpath) := (\ [\texttt{Jogador/Time}], \texttt{Time/Capitão})$
$\quad\quad in\ \underline{\texttt{/Time/Capitão}}$

Consulta reescrita: `/Time/Capitão`

■

4.5 Eliminação de Redundâncias

Após a aplicação da reescrita em uma consulta *CXPath* utilizando as dependências de inclusão, é necessário verificar se houve a inserção de redundâncias na consulta gerada. Por exemplo, considere o modelo conceitual da Figura 4.5 e a dependência de inclusão $\mathcal{ID} = \{$`Time.Capitão` $\subseteq$ `Jogador.Nome`$\}$. Quando aplicadas as funções de reescrita na consulta `/Jogador[Nome=X]/Time[Sigla=Y]` o seguinte resultado é obtido:

```
/Time[Capitão=X]/Time[Sigla=Y]
```

Pode-se observar que o conceito `Time` além de redundante está invalidando a consulta de acordo com o modelo conceitual. A consulta resultante mostra a navegação de conceito `Time` para ele mesmo, embora no modelo conceitual não exista um auto-relacionamento no conceito `Time`. Diante disso, nesta seção, é apresentado um conjunto de funções com o intuito de eliminar redundâncias possivelmente inseridas na aplicação do mecanismo de reescrita.

Um algoritmo para eliminar essas redundâncias deve:

- identificar conceitos iguais com relacionamentos iguais consecutivos em uma consulta (o tratamento de relacionamentos iguais já descarta a possibilidade do algoritmo reconhecer como redundância um auto-relacionamento);
- eliminar um dos relacionamentos juntamente com um dos conceitos e unir os predicados.

Outro tipo de redundância possivelmente inserida pelo mecanismo de reescrita é: considerando o modelo conceitual da Figura 4.4, a dependência de inclusão `A.x` $\subseteq$ `C.z` e a consulta `/C[z=1]/B/A[x=1]`.

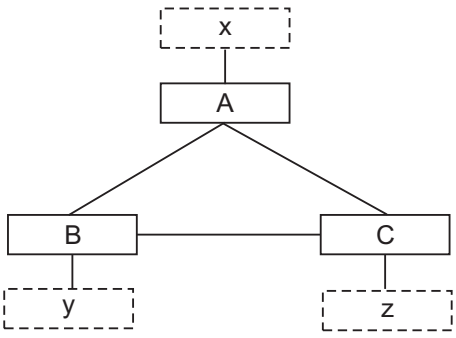


Figura 4.4: Modelo Conceitual Cíclico.

A consulta após o processo de reescrita seria `/A[x=1]/B/A[x=1]`. Claramente é identificada uma redundância nessa consulta gerada. Porém, tal tipo de redundância não é tratada nesta seção, pois embora redundante, a consulta está de acordo com o modelo conceitual, diferente das redundâncias mostradas anteriormente.

Exemplo 4.3 *Considere o modelo conceitual da Figura 4.5, a dependência de inclusão* $\mathcal{ID} = \{$`Time.Capitão` $\subseteq$ `Jogador.Nome`$\}$*, e a consulta CXPath abaixo,*

```
/Jogador[Nome=X]/Time[Sigla=Y]
```

perguntando por todos os times com sigla `Y` *no qual o jogador de nome* `X` *participa. Aplicando* Rew *sob essa consulta é obtido:*

`/Time[Capitão=X]/Time[Sigla=Y]`

Pode-se notar que foi inserida uma redundância na consulta gerada invalidando-a de acordo com o modelo conceitual. Identificada a redundância inserida, um dos conceitos é eliminado e os predicados dos conceitos redundantes são concatenados. Com isso, a consulta resultante seria:

`/Time[Capitão=X][Sigla=Y]`

Figura 4.5: Modelo Conceitual.

■

Dada como entrada uma consulta *CXPath* o mecanismo Rd gera uma nova consulta *CXPath* com a eliminação das redundâncias possivelmente inseridas no mecanismo de reescrita.

$$\text{Rd: CXPath} \rightarrow \text{CXPath}$$

O mecanismo de eliminação de redundâncias Rd é composto pelas funções $\mathbf{Rd_{APE}}$, $\mathbf{Rd_{RPE}}$ e $\mathbf{Rd_{PRE}}$. As funções $\mathbf{Rd_{APE}}$, $\mathbf{Rd_{RPE}}$ e $\mathbf{Rd_{PRE}}$ buscam a existência de redundâncias nas expressões de caminho absoluto, expressões de caminho relativo e nos predicados, respectivamente. Na função $\mathbf{Rd_{RPE}}$ (Subseção 4.5.2) ocorre o tratamento das redundâncias propriamente dito. Um dos relacionamentos e um dos conceitos redundantes é eliminado e os predicados são concatenados.

4.5.1 Eliminação de Redundâncias de Expressões Absolutas $(\mathbf{Rd_{APE}})$

A verificação de redundâncias em expressões de caminho absoluto é feita pela função $\mathbf{Rd_{APE}}$. Pela regra (RDAPE1) , como não existe nada para verificar redundância, a própria "/" é escrita como resultado.

(RDAPE1) $\mathbf{Rd_{APE}}(/) = /$

Na regra (RDAPE2) a barra é escrita na saída juntamente com aplicação da verificação de redundâncias em $RelPath$.

(RDAPE2) $\mathbf{Rd_{APE}}(/RelPath) = / \mathbf{Rd_{RPE}}(RelPath)$

4.5.2 Eliminação de Redundâncias de Expressões Relativas ($\mathbf{Rd_{RPE}}$)

A verificação de redundâncias em expressões de caminho relativo é feita pela função $\mathbf{Rd_{RPE}}$. Na regra (RDRPE1) como não são observados conceitos consecutivos, $Rel\ id$ é escrito na saída, e é feita a eliminação de redundâncias nos predicados $Preds$.

(RDRPE1) $\mathbf{Rd_{RPE}}(Rel\ id\ Preds) = Rel\ id\ \mathbf{Rd_{PRE}}(Preds)$

Quando forem observados apenas dois conceitos consecutivos na consulta, a regra utilizada é (RDRPE2) .

(RDRPE2) $\mathbf{Rd_{RPE}}(Rel_1\ id_1\ Preds_1\ /\ Rel_2\ id_2\ Preds_2) =$
$se\ \ (id_1 = id_2,\ Rel_1 = Rel_2)\ \ então$
$\mathbf{Rd_{RPE}}(Rel_2\ id_2\ \mathbf{Rd_{PRE}}(Preds_1\ Preds_2))$
$senão$
$Rel_1\ id_1\ \mathbf{Rd_{PRE}}(Preds_1)\ /\ Rel_2\ id_2\ \mathbf{Rd_{PRE}}(Preds_2)$

Se for identificada uma redundância $(id_1 = id_2,\ Rel_1 = Rel_2)$, o conceito id_1 juntamente com o relacionamento Rel_1 são eliminados e é aplicada recursivamente a função $\mathbf{Rd_{RPE}}$. Também é aplicada a função $\mathbf{Rd_{PRE}}$ para verificação de redundância na união dos predicados $Preds_1$ e $Preds_2$.

A regra (RDRPE3) funciona da mesma maneira que a regra (RDRPE2) . A única diferença é o tratamento de outras expressões de caminho relativo especificadas por $RelPath$.

(RDRPE3) $\mathbf{Rd_{RPE}}(Rel_1\ id_1\ Preds_1\ /\ Rel_2\ id_2\ Preds_2\ /\ RelPath) =$
$se\ \ (id_1 = id_2,\ Rel_1 = Rel_2)\ \ então$
$\mathbf{Rd_{RPE}}(Rel_2\ id_2\ \mathbf{Rd_{PRE}}(Preds_1\ Preds_2)\ /\ RelPath)$
$senão$
$Rel_1\ id_1\ \mathbf{Rd_{PRE}}(Preds_1)\ /\ \mathbf{Rd_{RPE}}(Rel_2\ id_2\ Preds_2\ /\ RelPath)$

4.5.3 Eliminação de Redundâncias em Predicados ($\mathbf{Rd_{PRE}}$)

A função $\mathbf{Rd_{PRE}}$ para verificação de redundância em predicados (regras (RDPRE1), (RDPRE2) e (RDPRE3)) conserva na saída os colchetes [ ], operadores op e os literais $Literal$. Além disso, na regra (RDPRE1) a expressão relativa $RelPath$ é avaliada pela função $\mathbf{Rd_{RPE}}$ e outros predicados são avaliados recursivamente com $\mathbf{Rd_{PRE}}$.

(RDPRE1) $\mathbf{Rd_{PRE}}([RelPath\ op\ Literal]\ Preds) =$
$[\ \mathbf{Rd_{RPE}}(RelPath)\ op\ Literal]\ \mathbf{Rd_{PRE}}(Preds)$

Na regra (RDPRE2) a expressão absoluta $/RelPath$ é avaliada pela função $\mathbf{Rd_{APE}}$ e outros predicados são avaliados com $\mathbf{Rd_{PRE}}$.

(RDPRE2) $\mathbf{Rd_{PRE}}([/RelPath\ op\ Literal]\ Preds) =$
$[\ \mathbf{Rd_{APE}}(/RelPath)\ op\ Literal]\ \mathbf{Rd_{PRE}}(Preds)$

Na regra (RDPRE3) é apenas retornado vazio, quando passado como parâmetro um predicado vazio.

(RDPRE3) $\mathbf{Rd_{PRE}}(\varepsilon) = \varepsilon$

Assim como na definição do mecanismo Rew de reescrita não foram consideradas aqui todas as possibilidades sintáticas da linguagem *CXPath* para os predicados. Novamente afirmamos que as regras que faltam seguem diretamente das regras já propostas.

Após o processo de eliminação de redundâncias é necessário a validação da consulta *CXPath* gerada com o modelo conceitual, processo definido em (SILVEIRA; HEUSER, 2007). Abaixo é apresentado um exemplo que mostra em detalhes a utilização do mecanismo de eliminação de redundâncias apresentado.

Exemplo 4.4 *Este exemplo considera a consulta* `/Sigla/Time/Time/Capitão` *gerada pela reescrita da consulta* `/Sigla/Time/Jogador/Nome` *através da dependência* `Time.Capitão` $\subseteq$ `Jogador.Nome`*. A aplicação das regras irá eliminar o conceito* `Time` *que está redundante, e neste caso, também está invalidando a consulta, pois* `Time` *não possui um auto-relacionamento no modelo apresentado na Figura 4.5.*

Para expressões de caminho absolutas são verificadas duas regras: (RDAPE1) *quando se trata apenas de uma "/" e* (RDAPE2) *quando se trata de uma "/"seguida de uma expressão relativa. Utilizando a regra* (RDAPE2) *na consulta obtém-se:*

$$\mathbf{Rd_{APE}}(\texttt{/Sigla/Time/Time/Capitão}) = \\ \texttt{/}\mathbf{Rd_{RPE}}(\varepsilon\ \texttt{Sigla}\ \varepsilon\ \texttt{/}\ \varepsilon\ \texttt{Time}\ \varepsilon\ \texttt{/Time/Capitão}) \tag{1}$$

Para tratar expressões relativas existem 3 regras. A regra utilizada nesse momento é a (RDRPE3) *pois a parte avaliada da consulta se trata de mais de duas expressões relativas seguidas, com o conceito* `Sigla` *diferente do conceito* `Time`*. Sendo assim, aplicando essa regra em (1):*

$$\mathbf{Rd_{APE}}(\texttt{/Sigla/Time/Time/Capitão}) = \\ \texttt{/}\ \varepsilon\ \texttt{Sigla}\ \mathbf{Rd_{PRE}}(\varepsilon)\ \texttt{/}\ \mathbf{Rd_{RPE}}(\varepsilon\ \texttt{Time}\ \varepsilon\ \texttt{/}\ \varepsilon\ \texttt{Time}\ \varepsilon\ \texttt{/}\ \texttt{Capitão}) \tag{2}$$

Para avaliar predicados existem 3 regras, porém a única que avalia predicados vazios é a regra (RDPRE3) *. Aplicando essa regra em (2) tem-se:*

$$\mathbf{Rd_{APE}}(\texttt{/Sigla/Time/Time/Capitão}) = \\ \texttt{/}\ \varepsilon\ \texttt{Sigla}\ \varepsilon\ \texttt{/}\ \mathbf{Rd_{RPE}}(\varepsilon\ \texttt{Time}\ \varepsilon\ \texttt{/}\ \varepsilon\ \texttt{Time}\ \varepsilon\ \texttt{/}\ \texttt{Capitão}) \tag{3}$$

Nesse momento, é utilizada uma das regras de expressões relativas para avaliar a expressão. Pode-se notar que a expressão é composta por mais de duas expressões relativas (de acordo com a gramática de CXPath), com o primeiro e segundo conceitos iguais, `Time`*, e dois relacionamentos também iguais,* ε*. A regra utilizada em (3) nesse caso é a* (RDRPE3)*, que elimina a redundância encontrada (elimina um dos conceitos* `Time`*) e aplica recursivamente a função* $\mathbf{Rd_{RPE}}$ *que trata de expressões relativas. Também é aplicada a função* $\mathbf{Rd_{PRE}}$ *na união dos predicados de cada um dos conceitos* `Time` *verificados.*

$$\mathbf{Rd_{APE}}(\texttt{/Sigla/Time/Time/Capitão}) = \\ \texttt{/}\ \varepsilon\ \texttt{Sigla}\ \varepsilon\ \texttt{/}\ \mathbf{Rd_{RPE}}(\varepsilon\ \texttt{Time}\ \mathbf{Rd_{PRE}}(\varepsilon)\ \texttt{/}\ \texttt{Capitão}) \tag{4}$$

Primeiramente em (4) é avaliado o predicado vazio pela regra (RDPRE3) *, para depois aplicar a função* $\mathbf{Rd_{RPE}}$*.*

$$\mathbf{Rd_{APE}}(\texttt{/Sigla/Time/Time/Capitão}) = \\ \texttt{/}\ \varepsilon\ \texttt{Sigla}\ \varepsilon\ \texttt{/}\ \mathbf{Rd_{RPE}}(\varepsilon\ \texttt{Time}\ \varepsilon\ \texttt{/}\ \varepsilon\ \texttt{Capitão}\ \varepsilon) \tag{5}$$

Para avaliar a expressão relativa de (5) é utilizada a regra (RDRPE2) *pois se trata de duas expressões relativas seguidas com conceitos diferentes.*

$$\begin{aligned}\mathbf{Rd_{APE}}(\texttt{/Sigla/Time/Time/Capitão}) = \\ \texttt{/}\,\varepsilon\,\texttt{Sigla}\,\varepsilon\,\texttt{/}\,\varepsilon\,\texttt{Time}\,\mathbf{Rd_{PRE}}(\varepsilon)\,\texttt{/}\,\varepsilon\,\texttt{Capitão}\,\mathbf{Rd_{PRE}}(\varepsilon)\end{aligned} \tag{6}$$

Por fim, em (6) é utilizada novamente a regra (RDPRE3) *para avaliar predicados vazios, e obter o resultado final da eliminação de redundâncias.*

$$\begin{aligned}\mathbf{Rd_{APE}}(\texttt{/Sigla/Time/Time/Capitão}) &= \texttt{/}\,\varepsilon\,\texttt{Sigla}\,\varepsilon\,\texttt{/}\,\varepsilon\,\texttt{Time}\,\varepsilon\,\texttt{/}\,\varepsilon\,\texttt{Capitão}\,\varepsilon \\ &= \texttt{/Sigla/Time/Capitão}\end{aligned}$$

■

Apresentados os mecanismos de tradução e reescrita de consultas, bem como o de eliminação de redundâncias, o Algoritmo 1 abaixo mostra em alto nível todo o processo de resolução de consultas até a obtenção das consultas *XPath* a serem submetidas às fontes.

Algoritmo 1 Resolução de consultas

Entrada: *Um sistema de integração* $\mathcal{I} = \langle \mathcal{G}, \mathcal{S}, \mathcal{M} \rangle$, *uma consulta global* Q_{cxpath}, *e um conjunto* $\mathcal{ID}$ *de dependências de inclusão*.

Saída: *Um conjunto de consultas* $\{XPath_1, ..., XPath_n\}$ *a ser aplicado nas fontes*.

1: $Q_{rew}, Q_{rd} \leftarrow \varnothing$
2: **for all** $inclusionDependency \in \mathcal{ID}$ **do**
3: $\quad Q_{rew} \leftarrow Q_{rew} \cup \mathsf{Rew}(Q_{cxpath}, inclusionDependency)$
4: **end for**

5: **for all** $q \in Q_{rew}$ **do**
6: $\quad Q_{rd} \leftarrow Q_{rd} \cup \mathsf{Rd}(q)$
7: **end for**

8: $Q_{xpath} \leftarrow \varnothing$
9: **for all** $q \in (Q_{cxpath} \cup Q_{rd})$ **do**
10: $\quad$ **for all** $M_i \in \mathcal{M}$ **do**
11: $\quad\quad Q_{xpath} \leftarrow Q_{xpath} \cup \mathsf{Tr}(q, M_i)$
12: $\quad$ **end for**
13: **end for**

14: **return** Q_{xpath}

A execução da resolução de consultas apresentada no Algoritmo 1 é feita da seguinte maneira:

i. Para cada dependência de inclusão pertencente ao conjunto $\mathcal{ID}$, a consulta *CXPath* é reescrita pelo mecanismo Rew;

ii. Para cada consulta reescrita é aplicado o mecanismo Rd de eliminação redundâncias;

iii. Por fim, a consulta original e as consultas geradas da etapa anterior são traduzidas para cada fonte do sistema de integração.

Pelo algoritmo de resolução de consultas apresentado, cada consulta passa necessariamente pelos mecanismos de reescrita e de eliminação de redundâncias.

4.6 Arcabouço Formal para Integração de Dados XML baseada em Modelos Conceituais com Dependências de Inclusão

Esta seção, assim como a Seção 2.6, apresenta o arcabouço formal para a abordagem para integração de dados utilizada nesta dissertação. A diferença é que nesta seção o arcabouço é mostrado com a adição de dependências de inclusão. Nessa seção, os esforços são concentrados nas mudanças ocasionadas pela inclusão das dependências no arcabouço mostrado na Seção 2.6.

Formalmente, um sistema de integração de dados $\mathcal{I}$ é uma tripla $\langle \mathcal{G}, \mathcal{S}, \mathcal{M} \rangle$, onde:

- $\mathcal{G}$ é o esquema global representado por um modelo conceitual juntamente com dependências de inclusão e chave. Em particular, $\mathcal{G} = \langle \mathcal{CM}, \mathcal{ID} \rangle$, onde

  i) $\mathcal{CM}$ representa o modelo conceitual, indicado por $\langle \mathcal{NL}, \mathcal{L}, \mathcal{R} \rangle$ (Definição 2.7);

  ii) $\mathcal{ID}$ é o conjunto de dependências de inclusão do tipo $nl_1.Rel_1\ l_1 \subseteq nl_2.Rel_2\ l_2$.

- $\mathcal{S}$ é o conjunto $\{S_1, S_2, ..., S_n\}$ de esquemas XML.

- $\mathcal{M}$ é o conjunto $\{M_1, M_2, ..., M_n\}$ de mapeamentos especificados entre $\mathcal{G}$ e $\mathcal{S}$, do tipo *global-as-view (GAV)*. Cada mapeamento M_i, sobre a fonte i, é formado por um conjunto de triplas do tipo $\langle g, q_{\mathcal{S}}, as \rangle$, onde g é um elemento do modelo conceitual, $q_{\mathcal{S}}$ é uma consulta *XPath* sobre a fonte i, e a suposição $as = sound$, especificando que as fontes podem ser incompletas.

Afim de determinar a semântica de um sistema de integração de dados $\mathcal{I}$ considere o conjunto de fontes $\mathcal{D} = \{D_1, D_2, ..., D_n\}$ de acordo com o conjunto de esquemas $\mathcal{S} = \{S_1, S_2, ..., S_n\}$. Baseado em $\mathcal{D}$ é especificado qual o conteúdo do esquema global $\mathcal{G}$. Chama-se de *banco de dados global* para $\mathcal{I}$ qualquer instância do esquema global $\mathcal{G}$, ou seja, qualquer banco de dados que atende as restrições de inclusão e cardinalidade impostas pelo esquema global.

Na Seção 2.6 as únicas restrições impostas pelo esquema global eram as cardinalidades impostas pelos relacionamentos. A definição de banco de dados global foi incrementada de forma a incorporar as dependências de inclusão (Definição 4.5).

Definição 4.5 *Uma base de dados $\mathcal{B}$ para um esquema global $\mathcal{G} = \langle \mathcal{CM}, \mathcal{ID} \rangle$, onde $\mathcal{CM}$ é um modelo conceitual e $\mathcal{ID}$ é um conjunto de dependências de inclusão, é definida como:*

- *Uma base de dados $\mathcal{B}$ conforme a Definição 2.9 e consistente em relação às dependências de inclusão $\mathcal{ID}$, ou seja, $\mathcal{B} \models \mathcal{ID}$.*

5 CONCLUSÃO

Neste trabalho foi apresentado o processo de tradução de consultas *CXPath*, expressas em relação a modelos conceituais, para consultas *XPath* expressas em relação aos esquemas de fontes XML de um sistema de integração. Esse processo de tradução foi definido indutivamente na estrutura sintática de consultas *CXPath* e é guiado pelas informações de mapeamento. Tais informações são geradas no processo de construção do modelo conceitual de forma a manter a correspondência entre elementos globais e locais (MELLO; HEUSER, 2005).

A seguir, foi considerado o processo de tradução levando em conta modelos conceituais mais expressivos nos quais é possível especificar dependências de inclusão. Pelo fato das fontes serem consideradas incompletas, a vantagem de usar modelos conceituais mais expressivos é a possibilidade de obter resultados de consultas mais completos.

Utilizando a informação sobre dependências de inclusão, uma consulta *CXPath*, submetida ao sistema de integração, passa antes por um processo de reescrita que produz consultas *CXPath* adicionais que capturam informações semânticas antes não expressas nos modelos conceituais. Esse processo de reescrita pode inserir redundâncias nas consultas *CXPath* resultantes, sendo assim, cada consulta reescrita passa também por um processo de verificação e eliminação de redundâncias. A consulta *CXPath* original e as consultas *CXPath* resultantes da reescrita, após a eliminação de redundâncias são, finalmente, traduzidas para consultas *XPath* nas fontes.

Além disso, a abordagem para integração de dados utilizada neste trabalho é apresentada utilizando o arcabouço formal proposto por (LENZERINI, 2002). Este arcabouço é bastante difundido na área de integração de dados e utilizado em diversos trabalhos (POGGI; ABITEBOUL, 2005; CALÌ et al., 2003; CALÌ et al., 2004; CALVANESE et al., 2001; LOPATENKO, 2004; EITER, 2005; LEONE et al., 2005). De acordo com o próprio autor, tal arcabouço é geral o bastante para capturar todas as abordagens de integração encontradas na literatura, o que inclui a abordagem para integração de dados utilizada nesta dissertação.

Um possível trabalho futuro é a prova de correção dos algoritmos propostos. Para tanto é necessário definir a semântica de consultas *CXPath* em relação a uma base de dados virtual, dada pelo modelo conceitual, e mostrar que o resultado da consulta *CXPath* sobre essa base virtual é um subconjunto ou equivalente ao resultado obtido pelas consultas *XPath* resultantes do processo de reescrita, eliminação de redundância e tradução.

A semântica de *CXPath* é necessária pois ofereceria uma definição precisa dos dados retornados de uma base virtual, dada pelo modelo conceitual, possibilitando a comparação dos resultados dessa base com os resultados das fontes. Como a abordagem para integração de dados utilizada nesta dissertação utiliza modelos de dados diferentes nos níveis global e local, não é possível uma comparação direta entre os resultados.

A formalização da tradução levando em consideração a decomposição de consultas também é outro possível trabalho futuro. Neste trabalho a formalização da tradução é feita considerando cada fonte de forma independente, ou seja, não há interação entre as fontes de forma a construir o resultado de uma consulta. A decomposição de consultas é uma característica natural quando se tratando em integrar dados de diferentes fontes. Sua formalização sistematizaria de maneira exata e de forma não ambígua a tradução de consultas que precisam ser decompostas sobre as fontes a fim de construir o resultado de uma consulta.

Outro possível trabalho é a utilização de mapeamentos do tipo $exact$ e verificar sua implicação no mecanismo de resolução de consultas. Para mapeamentos do tipo $sound$ é possível ter vários bancos de dados válidos para um sistema de integração de acordo com as fontes, pois as fontes podem ser incompletas. O mecanismo de reescrita utiliza a informação dos mapeamentos do tipo $sound$ justamente para buscar resultados possivelmente incompletos. Se todos os mapeamentos forem $exact$, cada mapeamento fornece exatamente o conjunto de dados que pode ser encontrado na extensão de um conceito global, ou seja, existirá apenas um banco de dados global válido. Nesse contexto não é necessária a reescrita de consultas.

O poder de expressão do modelo conceitual pode ser aumentado com outros tipos de dependências, como por exemplo, dependências de chave e exclusão, e estudada suas implicações na resolução de consultas. É necessário um estudo sobre essas dependências pois sua utilização pode deixar a resolução de consultas sem sentido ($sem(\mathcal{I}, \mathcal{D}) = \emptyset$), ou seja, os dados são inconsistentes com a especificação do esquema global.

Em um sistema de integração, os dados, em geral, são provenientes de fontes autônomas, cada fonte com sua construção e evolução distintas. Por essa razão, os dados contidos nas fontes podem não respeitar as restrições impostas pelo esquema global. Sendo assim, outro trabalho futuro seria o tratamento dessas inconsistências. Por exemplo, se em um esquema global o CPF identifica unicamente uma pessoa, e nas fontes existem duas pessoas diferentes especificadas com o mesmo CPF, qual resposta deve ser retornada?

As inconsistências são geralmente tratadas através de procedimentos de transformação que devem ser aplicados aos dados retornados de uma consulta de forma a torná-la consistente (GALHARDAS et al., 2000). Apesar do tratamento de inconsistências ser importante, ele é geralmente ignorado na maior parte dos sistemas de integração de dados.

No algoritmo de resolução de consultas apresentado, seria interessante antes da aplicação dos mecanismos, verificar a necessidade de reescrita da consulta e de eliminação de redundâncias. No algoritmo mostrado poderiam ser inseridas funções como $needsRewrite$ e $hasRedundancy$ para tal propósito. Porém, é necessário verificar a viabilidade dessas funções, ou seja, se a inclusão delas no algoritmo não degradará o desempenho do sistema. O Algoritmo 2 apresenta o processo de resolução de consultas com a inclusão das funções $needsRewrite$ (linha 3), e $hasRedundancy$ (linha 8).

Algoritmo 2 Resolução de consultas

Entrada: *Um sistema de integração* $\mathcal{I} = \langle \mathcal{G}, \mathcal{S}, \mathcal{M} \rangle$, *uma consulta global* Q_{cxpath}, *e um conjunto* $\mathcal{ID}$ *de dependências de inclusão*

Saída: *Um conjunto de consultas* $\{XPath_1, ..., XPath_n\}$ *a ser aplicado nas fontes.*

$Q_{rew}, Q_{rd} \leftarrow \varnothing$
for all $inclusionDependency \in \mathcal{ID}$ **do**
    if $(needsRewrite(Q_{cxpath}, inclusionDependency) ==$ **true**$)$ **then**
        $Q_{rew} \leftarrow Q_{rew} \cup \mathsf{Rew}(Q_{cxpath}, inclusionDependency)$
    end if
end for

for all $q \in Q_{rew}$ **do**
    if $(hasRedundancy(q) ==$ **true**$)$ **then**
        $Q_{rd} \leftarrow Q_{rd} \cup \mathsf{Rd}(q)$
    else
        $Q_{rd} \leftarrow Q_{rd} \cup q$
    end if
end for

$Q_{xpath} \leftarrow \varnothing$
for all $q \in (Q_{cxpath} \cup Q_{rd})$ **do**
    for all $M_i \in \mathcal{M}$ **do**
        $Q_{xpath} \leftarrow Q_{xpath} \cup \mathsf{Tr}(q, M_i)$
    end for
end for

return Q_{xpath}

REFERÊNCIAS

ABITEBOUL, S.; HULL, R.; VIANU, V. **Foundations of Databases**. [S.l.]: Addison-Wesley, 1995.

ABITEBOUL, S.; SEGOUFIN, L.; VIANU, V. Representing and querying XML with incomplete information. **ACM Trans. Database Syst.**, [S.l.], v.31, n.1, p.208–254, 2006.

AMANN, B. et al. Querying XML Sources Using an Ontology-Based Mediator. In: COOPIS/DOA/ODBASE, 2002, London, UK. **Proceedings...** Berlin: Springer-Verlag, 2002. p.429–448.

CALì, A. et al. IBIS: Semantic Data Integration at Work. In: INTERNATIONAL CONFERENCE ON ADVANCED INFORMATION SYSTEMS ENGINEERING, CAISE, 15., 2003. **Proceedings...** Berlin: Springer, 2003. p.79–94. (Lecture Notes in Computer Science, v.2681).

CALÌ, A.; LEMBO, D.; ROSATI, R.; RUZZI, M. Experimenting Data Integration with DIS@DIS. In: INTERNATIONAL CONFERENCE ON ADVANCED INFORMATION SYSTEMS ENGINEERING, CAISE, 16., 2004, Riga, Latvia. **Proceedings...** Berlin: Springer, 2004. p.51–66. (Lecture Notes in Computer Science, v.3084).

CALÌ, A. et al. Accessing Data Integration Systems through Conceptual Schemas. In: INTERNATIONAL CONFERENCE ON CONCEPTUAL MODELING, ER, 20., 2001, Yokohama. **Conceptual Modeling:** proceedings. Berlin: Springer-Verlag, 2001. p.270–283. (Lecture Notes in Computer Science, v.2224).

CALÌ, A. et al. Query Rewriting and Answering under Constraints in Data Integration Systems. In: INTERNATIONAL JOINT CONFERENCE ON ARTIFICIAL INTELLIGENCE, IJCAI, 18., 2003, Acapulco. **Proceedings...** San Francisco: Morgan Kaufmann, 2003. p.16–21.

CALVANESE, D. et al. A Framework for Ontology Integration. In: SWWS, 2001. **Proceedings...** [S.l.: s.n.], 2001. p.303–316.

CAMILLO, S. D.; HEUSER, C. A.; MELLO, R. S. Querying Heterogeneous XML Sources through a Conceptual Schema. In: INTERNATION CONFERENCE ON CONCEPTUAL MODELING, ER, 22., 2003. **Conceptual Modeling:** proceedings. Berlin: Springer-Verlag, 2003. p.186–199. Conceptual Modeling.

DEUTSCH, A.; POPA, L.; TANNEN, V. Physical Data Independence, Constraints, and Optimization with Universal Plans. In: VLDB CONFERENCE, 25., 1999. **Proceedings...** [S.l.: s.n.], 1999. p.459–470.

DEUTSCH, A.; POPA, L.; TANNEN, V. Query reformulation with constraints. **SIGMOD Record**, [S.l.], v.35, n.1, p.65–73, 2006.

EITER, T. Data Integration and Answer Set Programming. In: INTERNATIONAL CONFERENCE ON LOGIC PROGRAMMING AND NONMONOTONIC REASONING, LPNMR, 8., 2005, Diamante, Italy. **Proceedings...** Berlin: Springer, 2005. p.13–25. (Lecture Notes in Computer Science, v.3662).

FAN, W. et al. On XML Integrity Constraints in the Presence of DTDs. In: SYMPOSIUM ON PRINCIPLES OF DATABASE SYSTEMS, PODS, 20., 2001. **Proceedings...** [S.l.: s.n.], 2001.

FAN, W. et al. Secure XML Querying with Security Views. In: ACM SIGMOD INTERNATIONAL CONFERENCE ON MANAGEMENT OF DATA, SIGMOD, 2004, Paris, France. **Proceedings...** New York: ACM, 2004. p.587–598.

FEIJÓ, D. V.; FUZITAKI, C. N.; MOREIRA, A.; GALANTE, R. M.; HEUSER, C. A. CXPath: A Query Language for Conceptual Models of Integrated XML Data. In: INTERNATIONAL CONFERENCE ON SOFTWARE ENGINEERING AND KNOWLEDGE, SEKE, 19., 2007, Boston, Massachusetts, USA. **Proceedings...** Skokie: Knowledge Institute, 2007. p.592–597.

GALHARDAS, H. et al. An Extensible Framework for Data Cleaning. In: INTERNATION CONFERENCE ON DATA ENGINEERING, ICDE, 16., 2000, San Diego, CA. **Proceedings...** Los Alamitos: CA: IEEE Computer Society, 2000. p.312.

HAAS, L. M. Beauty and the Beast: The Theory and Practice of Information Integration. In: INTERNATIONAL CONFERENCE ON DATABASE THEORY, ICDT, 11., 2007. **Proceedings...** [S.l.: s.n.], 2007. p.28–43.

HALEVY, A. Answering Queries Using Views: A Survey. **VLDB Journal**, [S.l.], p.270–294, 2001.

HALPHIN, T. **Object-Role Modeling (ORM/NIAM), Handbook on Architectures of Information Systems**. [S.l.]: Berlin, Springer - Verlag, 1998. p.81–102.

LENZERINI, M. Data Integration: A Theoretical Perspective. In: SYMPOSIUM ON PRINCIPLES OF DATABASE SYSTEMS, PODS, 21., 2002. **Proceedings...** New York: ACM, 2002. p.233–246.

LEONE, N. et al. The INFOMIX System for Advanced Integration of Incomplete and Inconsistent Data. In: ACM SIGMOD INTERNATIONAL CONFERENCE ON MANAGEMENT OF DATA, SIGMOD, 2005. **Proceedings...** New York: ACM, 2005. p.915–917.

LOPATENKO, A. Query Answering Under Exact View Assumption in Local As View Data Integration System. In: INTERNATIONAL CONFERENCE ON PRINCIPLES OF KNOWLEDGE REPRESENTATION AND REASONING, KR, 9., 2004. **Proceedings...** [S.l.: s.n.], 2004.

MANOLESCU, I. et al. Answering XML Queries on Heterogeneous Data Sources. In: VLDB, 2001. **Proceedings...** [S.l.: s.n.], 2001. p.241–250.

MELLO, R. S. **Uma Abordagem *Bottom-Up* para a Integração Semântica de Esquemas XML**. 2002. Tese (Doutorado em Ciência da Computação) — Instituto de Informática, Universidade Federal do Rio Grande do Sul, Porto Alegre.

MELLO, R. S.; HEUSER, C. A. BInXS: A Process for Integration of XML Schemata. In: CAISE, 17., 2005. **Proceedings...** Belin: Springer, 2005. p.151–166.

POGGI, A.; ABITEBOUL, S. XML Data Integration with Identification. In: DBPL, 2005. **Proceedings...** [S.l.: s.n.], 2005. p.106–121.

POPA, L. et al. Translating Web Data. In: VLDB, 2002. **Proceedings...** [S.l.: s.n.], 2002. p.598–609.

SILVEIRA, F. V. **Fragmentação e Decomposição de Consultas em XML**. 2006. Dissertação (Mestrado em Ciência da Computação) — Instituto de Informática, Universidade Federal do Rio Grande do Sul, Porto Alegre.

SILVEIRA, F. V.; HEUSER, C. A. A Two Layered Approach for Querying Integrated XML Sources. In: INTERNATIONAL DATABASE ENGINEERING AND APPLICATIONS SYMPOSIUM, IDEAS, 11., 2007. **Proceedings...** Los Alamitos: IEEE Computer Society, 2007. p.3–11.

ULLMAN, J. D. Information Integration Using Logical Views. **Theoretical Computer Science**, Amsterdam, v.239, n.2, p.189–210, 2000.

W3 Schools - The best things in life are free. Disponível em: <*http://www.w3schools.com*>. Acesso em: julho 2008.

W3C: the world wide web consortium. Disponível em: <*http://www.w3.org*>. Acesso em: julho 2008.

W3C XML Schema. Disponível em: <*http://www.w3.org/XML/Schema*>. Acesso em: julho 2008.

XML Path Language (XPath) Version 1.0 - W3C Recomendation 16 November 1999. Disponível em: <*http://www.w3.org/TR/xpath*>. Acesso em: agosto 2007.

XQUERY 1.0: An XML Query Language - W3C Recomendation 23 January 2007. Disponível em: <*http://www.w3.org/TR/xquery*>. Acesso em: agosto 2007.

XU, W.; ÖZSOYOGLU, Z. M. Rewriting XPath Queries Using Materialized Views. In: VLDB, 2005. **Proceedings...** [S.l.: s.n.], 2005. p.121–132.

YU, C.; POPA, L. Constraint-Based XML Query Rewriting For Data Integration. In: ACM SIGMOD INTERNATIONAL CONFERENCE ON MANAGEMENT OF DATA, SIGMOD, 2004. **Proceedings...** New York: ACM, 2004. p.371–382.

APÊNDICE A XML E XPATH

O objetivo deste apêndice é introduzir as noções básicas de XML e *XPath*, que são utilizados ao longo desta dissertação. O conteúdo aqui apresentado é baseado nas informações presentes em (W3 SCHOOLS - THE BEST THINGS IN LIFE ARE FREE, 2008).

XML

XML (*eXtensible Markup Language*) é uma linguagem de marcação desenvolvida pelo W3C (W3C: THE WORLD WIDE WEB CONSORTIUM, 2008) principalmente para solucionar as limitações de interoperabilidade e escalabilidade na Web impostas pela HTML. Seu principal objetivo é facilitar o compartilhamento de dados estruturados entre diferentes sistemas de informação, particularmente através da Internet.

No mundo real, sistemas de computadores e bancos de dados contém dados em formatos geralmente incompatíveis. Dados XML são armazenados em formato texto. Isso provê uma maneira independente de software e hardware de armazenar dados, o que torna muito mais fácil criar dados que diferentes aplicações podem compartilhar.

XML não surgiu com o intuito de substituir HTML. XML e HTML foram desenvolvidas com diferentes objetivos: XML foi desenvolvida para transportar e armazenar dados, com foco no que os dados representam. HTML foi desenvolvida com foco na apresentação dos dados.

Os rótulos utilizados em HTML (e na estrutura de HTML) são pré-definidos. Documentos HTML podem apenas utilizar rótulos definidos no padrão HTML (como <p>, <h1>, etc.). XML permite ao autor do documento definir seus próprios rótulos e sua própria estrutura para o documento.

O termo *documento XML* é utilizado para se referir a arquivos XML. Um documento XML possui elementos XML. Um elemento é constituído por um **rótulo de início**, um **conteúdo** e um **rótulo de fim**. Um elemento pode conter em seu conteúdo outros elementos, apenas texto, ou uma mistura dos dois. Elementos podem também possuir atributos. A Figura A.1 apresenta um documento XML contendo informações sobre artigos e autores.

Um **elemento composto** é um elemento com atributos e/ou um elemento que possui outros elementos em seu conteúdo. Na Figura A.1, são exemplos de elementos compostos: `publications`, `article`, e `author`. Um **elemento simples** é um elemento que possui apenas texto em seu conteúdo, por exemplo, o elemento `name`.

Um **elemento vazio** não possui conteúdo, ou seja, seu conteúdo é vazio. Elementos vazio são representados apenas pelo rótulo <.../>. A Figura A.2 apresenta um elemento `author` contendo um elemento `name` vazio. Um **elemento misto** possui outros elemen-

```
<publications>
   <article title= "Answering Regular Path Queries Using Views" year= "2000">
      <author>
         <name> Diego Calvanese </name>
      </author>
      <author>
         <name> Andrea Calì </name>
      </author>
      <author>
         <name> Guiseppe de Giacomo </name>
      </author>
      <author>
         <name> Maurizio Lenzerini </name>
      </author>
   </article>
   <article title= "Data Integration: A Theoretical Perspective" year= "2002">
         <author>
         <name> Maurizio Lenzerini </name>
      </author>
   </article>
</publications>
```

Figura A.1: Exemplo de documento XML.

```
...
<author>
   <name/>
</author>
...
```

Figura A.2: Exemplo de elemento vazio.

tos e texto em seu conteúdo. Na Figura A.3 é apresentado o elemento misto `author` contento texto e o elemento simples `name`.

Um **atributo** descreve uma propriedade de um elemento. Seu valor é especificado no rótulo de início de um elemento. Os valores dos atributos devem sempre ser especificados entre aspas duplas ou aspas simples. Na Figura A.1 `title` e `year` são atributos do elemento `article`.

Documentos XML formam uma estrutura de árvore que começa pela *raíz* e se ramifica até as *folhas*. No documento apresentado na Figura A.1 o elemento raíz é representado pelo elemento `publications`. As folhas são representadas pelos elementos simples `name`.

Documentos XML devem possuir um **elemento raíz**. Esse elemento é *pai* de todos os outros elementos. Os elementos de um documento XML podem ter sub-elementos, esses são chamados **elementos filhos**. Os termos pai, filho, irmão, são utilizados para descrever o relacionamento entre elementos. **Elementos pai** possuem filhos. Filhos no mesmo nível

```
...
<author>
   The article's author
   <name> Benjamin Pierce </name>
   is a good researcher
</author>
...
```

Figura A.3: Exemplo de elemento misto.

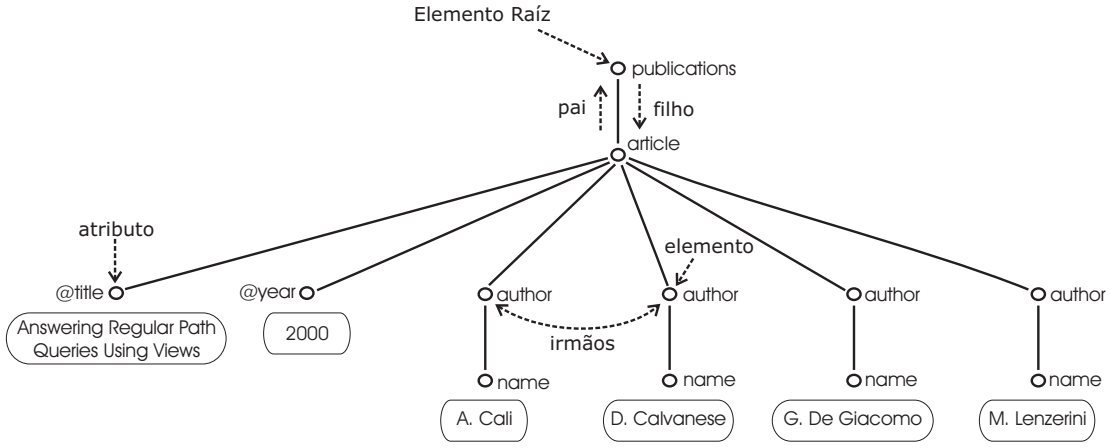


Figura A.4: Representação de um artigo da Figura A.1.

são chamados irmãos (Figura A.4).

O elemento raíz na Figura A.1 é `publications`. Todos elementos `article` estão contidos no elemento `publications`. O elemento `article` possui dois atributos, `title` e `year`, e um filho, `author`.

Um esquema define as construções legais de um documento XML. Ele define a estrutura de um documento com uma lista de atributos e elementos permitidos. Um elemento no esquema pode possuir uma seqüência e/ou escolhas de elementos e atributos permitidos. Os símbolos `?`, `*`, `+` utilizados na construção de expressões regulares são utilizados na definição de esquemas XML. Os operadores de expressões regulares apresentados indicam o número de ocorrências de um elemento em um documento XML, denotando, respectivamente, 0 ou 1 ocorrência, 0 ou n ocorrências, e 1 ou n ocorrências.

A Figura A.5 apresenta um esquema XML para a Figura A.1. Os elementos sem a representação dos operadores de expressões regulares indicam exatamente uma ocorrência do elemento. O elemento `publications` possui 0 ou mais ocorrências de `article`, este por sua vez possui 1 ou mais elementos `author`. Elementos iniciados com @ representam atributos, por exemplo, `title` e `year`.

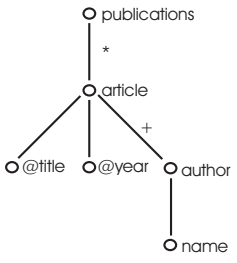


Figura A.5: Exemplo de esquema XML.

A Figura A.5 é apenas uma representação abstrata de um esquema XML especificado através de uma DTD (*Document Type Definition*) ou um XSD (*XML Schema Definition*) (W3C XML SCHEMA, 2008).

XPath

XPath (XML Path Language) é parte importante da família de tecnologias XML, já que ela provê a possibilidade de selecionar e filtrar dados contidos em um ou mais documentos XML. A sintaxe adotada pelo *XPath* é bastante intuitiva, utilizando expressões de caminho, assim como os caminhos de um sistema de arquivos, para selecionar elementos de um documento XML. Considere, por exemplo, a expressão *XPath* a seguir:

```
/publications/article
```

Se aplicada ao documento XML da Figura A.1, esta expressão retornará todos os elementos `article` que são filhos do elemento `publications`. Note que a expressão inicia com um caractere de barra (/). Toda expressão que inicia com uma barra indica que a pesquisa deve começar a partir da raiz do documento XML.

Vários caracteres de barra podem aparecer em uma expressão *XPath*. No exemplo acima, além da barra inicial, temos mais uma barra separando `publications` de `article`. Cada caractere de barra em uma expressão *XPath* representa uma troca de nível hierárquico. Com estas informações em mãos, pode-se concluir que o elemento `publications` é o elemento raiz do documento de exemplo e que ele é pai de do elemento `article`.

As expressões de caminho mais utilizadas em *XPath* são listadas abaixo:

Expressão	Descrição
/	Seleciona nós a partir da raiz
//	Seleciona nós no documento XML não importando sua posição
..	Seleciona o pai do nó atual
@	Seleciona atributos

A tabela abaixo apresenta algumas expressões *XPath*, utilizando o documento XML da Figura A.1, e seus respectivos resultados.

Exemplo	Resultado
`/publications`	Seleciona o nó raiz `publications`
`publications/article`	Seleciona todos os nós `article` que são filhos de `publications`.
`//article`	Seleciona todos os nós `article` não importando sua posição no documento.
`publications//author`	Seleciona todos nós `author` que são descendentes de `publications` não importando sua posição abaixo de `publications`.
`article/author/..`	Seleciona os nós pai do elemento `author`
`//@title`	Seleciona todos os atributos de nome `title`

Uma expressão de caminho pode ser absoluta ou relativa. Uma expressão de caminho absoluta inicia com uma barra (/) e uma expressão de caminho relativa não. Em ambos os casos uma expressão de caminho consiste na navegação entre elementos, cada um separado por uma barra. A expressão *XPath* abaixo representa uma expressão de caminho absoluta:

```
/publications/article/author
```

A expressão *XPath* abaixo representa uma expressão de caminho relativa, que retorna os nomes dos autores relacionados com artigos:

```
article/author/name
```

A linguagem *XPath* também permite a especificação de condições que tornam possível a filtragem dos dados. A expressão abaixo retorna um elemento de artigo específico, baseado numa condição de filtragem:

```
/publications/article[@year ='2007']
```

A expressão entre colchetes, formalmente denominada "predicado", determina uma condição lógica que deve ser atendida para que o nó que a precede seja selecionado pelo processador *XPath*. No exemplo acima, somente serão retornados os elementos `article` cujo atributo, `year`, tenha o valor `2007`. Os nós que aparecem no predicado da expressão estão dentro de um contexto; e este contexto é justamente o nó que precede o predicado. Na expressão acima, o nó de contexto do elemento `year` é o elemento `article`.

Note que um atributo deve ser precedido por um caractere `@`. Isso é suficiente para que o processador *XPath* saiba que está sendo feita referência para um atributo e não para um elemento.