3
Problema de decisão em esquemas conceituais

Um sistema de representação do conhecimento baseado em Lógica de Descrição (LD) é capaz de executar tipos específicos de inferência lógica. O domínio da aplicação é modelado construindo-se uma terminologia contendo definições de conceitos. Durante este processo, é importante descobrir se a terminologia permanece satisfatível, ou seja, se existe pelo menos uma interpretação s que satisfaz os axiomas da terminologia, o que equivale a dizer que existe um modelo para a terminologia (veja Seção 2.1).

Testar satisfatibilidade de terminologias é um problema básico, pois vários outros problemas podem a ele ser reduzidos. Em particular, o problema de subsunção reduz-se ao problema de satisfatibilidade em famílias de LD suficientemente expressivas [4]. O *problema de subsunção* em lógica de descrição refere-se à questão de decidir se uma expressão sempre denota um subconjunto do conjunto definido por outra expressão.

Este capítulo explora reduções do problema de inclusão de consultas e problemas semelhantes ao problema de subsunção em LD [29]. Dentre os problemas tratados, situa-se o problema da definição do conjunto de restrições de um esquema importado a partir do conjunto de restrições do esquema exportado e do mapeamento entre os dois esquemas. Adota-se em todo o capítulo os esquemas ultra elementares apresentados na Seção 2.3. Esta família de esquemas configura-se como um dialeto de LD e é suficientemente expressiva para cobrir várias classes de restrições de integridade e expressões de consulta. O problema de subsunção para o dialeto considerado é abordado de duas formas: tanto modificando-se o procedimento de decisão tradicional baseado em tableau [4], quanto um procedimento estrutural de decisão bem conhecido [21, 34] para considerar as restrições dos esquemas ultra elementares (um procedimento estrutural, como o nome indica, explora diretamente a estrutura sintática das expressões).

Este capítulo está organizado da seguinte forma. A Seção 3.1 explora reduções do problema de inclusão de consultas e problemas semelhantes ao problema de subsunção em LD. Em seguida, a Seção 3.2 descreve como modificar o procedimento de decisão baseado em tableau para tratar as restrições dos esquemas ultra elementares. Por fim, a Seção 3.3 apresenta um procedimento estrutural para o problema de subsunção no dialeto adotado.

3.1.
Inclusão de consultas e problemas de decisão correlatos

Inclusão de consultas refere-se ao problema de decidir se uma consulta Q sempre retorna um subconjunto do resultado de uma segunda consulta Q'. Um caso especial importante do problema de inclusão de consultas é chamado de *problema de consulta eficaz* e testa se uma consulta Q sempre retorna um resultado vazio.

Outro problema relacionado é chamado de *problema de restrição de subconjunto do esquema importado*. Recorde da Seção 2.2, que as restrições de subconjunto de um esquema exportado são traduzidas para restrições de subconjunto de um esquema importado. O problema em questão diz respeito então a computar quais, e como, as restrições de subconjunto do esquema exportado são traduzidas para as restrições do esquema importado, respeitando os mapeamentos de esquema.

Recorde que o problema de subsunção em lógica de descrição refere-se à questão de decidir se uma descrição de conceito sempre denota um subconjunto de um conjunto denotado por outra descrição de conceito [39]. O problema de subsunção é decidível para dialetos expressivos de LD, mas tipicamente pertence às classes de complexidade difíceis, especialmente na presença dos axiomas, como melhor explorado na Seção 1.4 de trabalhos relacionados.

Note que os três problemas citados acima podem ser reduzidos ao problema de subsunção. Porém, qualquer procedimento de decisão prático para estes problemas deve levar em conta as restrições de integridade do esquema exportado (ou seja, os axiomas da teoria) e ser eficiente. De fato, este é um dos desafios tratados nesta tese.

No contexto de esquemas ultra elementares, os problemas mencionados acima podem ser formalizados como se segue:

1. *O problema de inclusão de consulta*

   Dado um esquema ultra elementar $S=(\mathcal{L}, \Sigma)$ e duas descrições de conceito C e D de $\mathcal{L}$, determine se $\Sigma \models C \sqsubseteq D$.

2. *O problema de consulta eficaz*

 37

Dado um esquema ultra elementar $\mathcal{S}=(\mathcal{L}, \Sigma)$ e uma descrição de conceito C de $\mathcal{L}$, determine se $\Sigma \models C \sqsubseteq \bot$

3. *O problema de restrições de subconjunto do esquema importado*[1]
Dado um esquema externo ultra elementar $\mathcal{S}=(\mathcal{L},\Sigma)$, um esquema importado $I=(\mathcal{K}, \Gamma)$ sobre $\mathcal{S}$, e duas classes, C e D, de $\mathcal{K}$ com definições $C \equiv \mu_C$ e $D \equiv \mu_D$, determinar se $\Sigma \models \mu_C \sqsubseteq \mu_D$.

Intuitivamente, para o problema (1) de inclusão de consultas, C e D capturam consultas sobre $\mathcal{S}$, que podem ser fragmentos da mesma consulta ou de consultas diferentes. Se for possível provar que $\Sigma \models C \sqsubseteq D$, então a consulta C sempre irá retornar um subconjunto da consulta D, na presença das restrições de integridade Σ. O problema (2) é um caso especial do problema (1). Se for possível provar que $\Sigma \models C \sqsubseteq \bot$, então a consulta C sempre irá retornar um conjunto vazio (que é a interpretação padrão do conceito $\bot$), na presença das restrições de integridade Σ. Já o problema (3), pode ser entendido da seguinte forma: assuma que $\Sigma \models \mu_C \sqsubseteq \mu_D$. Já que $C \equiv \mu_C$ e $D \equiv \mu_D$, tem-se que $\boldsymbol{v} \models C \sqsubseteq D$, para qualquer interpretação $\boldsymbol{v}$ de I induzida por uma interpretação de $\mathcal{L}$ que satisfaça Σ.

Quaisquer desses problemas admitem variantes dependendo no dialeto de LD escolhido para escrever as descrições de conceito. Esta observação é de significância prática uma vez que, como já foi dito, a complexidade dos problemas depende da escolha do dialeto. Ressalta-se que, no contexto deste capítulo, o dialeto adotado corresponde a esquemas ultra elementares.

Para todos os exemplos deste capítulo será usado o ambiente de mediação *Sales* definido no Exemplo 2-2 usando apenas o esquema da *Amazon*, cujo resumo é apresentado na

Figura 15 (página 33) ignorando as restrições de cardinalidade definidas. No entanto, para tornar o exemplo mais interessante, serão considerados, para o ambiente de mediação, apenas produtos cujo título seja definido. Da mesma forma, somente os livros que tenham título e editora definidos. Esta pequena modificação implica em um novo mapeamento local do esquema exportado E_a da *Amazon* para o esquema importado I_a. De fato, o mapeamento local γ_a da

[1] Veja Seção 2.4 onde é introduzido o conceito de esquema importado.

 38

Figura 9 é substituído pelo mapeamento local $\gamma_a{}'$ da Figura 16, o que não gera modificações nas demais definições do ambiente de mediação *Sales*.

ai:Product ≡ a:Product ⊓ ∃a:title ai:Music ≡ a:Music ai:Book ≡ a:Book ⊓ ∃a:title ⊓ ∃a:pub	ai:title ≡ a:title ai:city ≡ a:pub∘a:city

Figura 16 – Novo Mapeamento Local $\gamma_{a'}$

Exemplo 3-1 – Inclusão de consulta

Primeiramente, para o problema de inclusão de consultas, considere o esquema ultra elementar $S = (\mathcal{A}, \Sigma)$ da Figura 15 com o novo mapeamento local $\gamma_a{}'$ da Figura 16, e as seguintes descrições de conceito:

(1) $Q_1 \equiv$ `a:Product`

(o conjunto de todos os produtos)

(2) $Q_2 \equiv$ `a:Book ⊓ ∃a:author`

(o conjunto dos livros que possuem autor conhecido)

(3) $Q_3 \equiv$ `a:Book ⊓ ∀a:author.{'Shakespeare'}`

(o conjunto de livros que, se o autor é conhecido, é `'Shakespeare')`

Pode-se facilmente ver que $\Sigma \models Q_2 \sqsubseteq Q_1$ e que $\Sigma \models Q_3 \sqsubseteq Q_1$, uma vez que `a:Book ⊑ a:Product` está em Σ. No entanto, não se pode provar que $\Sigma \models Q_3 \sqsubseteq Q_2$ pois não existe nenhum axioma em Σ garantindo que todos os livros possuem apenas autores conhecidos. Sem este axioma, a resposta à consulta Q_3 pode conter livros cujo autor é desconhecido devido ao quantificador $\forall$.

Exemplo 3-2 – Consulta Eficaz

As duas próximas descrições de conceito consideram o problema de consulta eficaz ainda no esquema $S = (\mathcal{A}, \Sigma)$

(4) $Q_4 \equiv$ `a:Book ⊓ ¬a:Product`

(o conjunto de todos os livros que não são produtos)

(5) $Q_5 \equiv$ `a:Book ⊓ ∀a:author.{'Shakespeare'} ⊓ ∃a:author.{'Marlowe'}`

(o conjunto de livros que, se o autor é conhecido, ele é `'Shakespeare'` ao mesmo tempo em que é `'Marlowe')`

 39

É possível mostrar que $\Sigma \models Q_4 \sqsubseteq \bot$ tomando novamente como base que o axioma `a:Book` $\sqsubseteq$ `a:Product` pertencente a Σ. De fato, $\models Q_5 \sqsubseteq \bot$, já que, por definição, constantes diferentes denotam indivíduos diferentes, devido à suposição de unicidade apresentada na Seção 2.1.

Exemplo 3-3 – Restrições de subconjunto do esquema importado

Este último exemplo ilustra o exemplo do problema das restrições de subconjunto do esquema importado. Considere novamente o esquema exportado ultra elementar $S = (\mathcal{A}, \Sigma)$ e o esquema importado $I=(\mathcal{K}, \Gamma)$ da Figura 15 com o novo mapeamento local $\gamma_a{}'$ da Figura 16.

Neste exemplo, deseja-se provar que $\Sigma \models$ `ai:Book` $\sqsubseteq$ `ai:Product`. Recorde que os mapeamentos incluem:

(6) `ai:Product` $\equiv$ `a:Product` $\sqcap$ $\exists$`a:title`

(7) `ai:Book` $\equiv$ `a:Book` $\sqcap$ $\exists$`a:title` $\sqcap$ $\exists$`a:pub`

Como $\Sigma \models$ `a:Book` $\sqsubseteq$ `a:Product`, segue então que

(8) $\Sigma \models$ `ai:Book` $\sqsubseteq$ `ai:Product`

Logo, é possível provar que $\Sigma \models$ `ai:Book` $\sqsubseteq$ `ai:Product`. O Exemplo 3-4 abordará com mais detalhes este problema.

3.2.
Extensão do tableau para esquemas ultra elementares

Nesta seção será estudado como estender o procedimento de decisão de tableau para o problema de subsunção [4] de tal forma a levar em conta as restrições de domínio, de imagem e de subconjunto. Tal extensão é, de fato, um procedimento de decisão para os problemas definidos na Seção 3.1, quando as descrições de conceito e de papel consideradas podem usar quaisquer das construções definidas na Seção 2.1.

Os tableaux são construídos utilizando-se as regras usuais [4] e novas regras, resumidas na Figura 17. Estas regras não recorrem a reduções para codificar as restrições de integridade, que é um procedimento pouco prático. Realmente, como observado em [19], pelo uso de união e fecho transitivo de papéis, pode-se codificar um conjunto de inclusões $\Sigma =\{C_1 \sqsubseteq D_1,..., C_n \sqsubseteq D_n\}$ como um conjunto de descrições de conceito $\Sigma^*=\{\neg C_1 \sqcup D_1 ,..., \neg C_n \sqcup D_n\}$ de modo que uma descrição de conceito d é satisfatível com respeito a Σ sse a seguinte descrição de conceito for satisfatível:

$$d \sqcap \forall(R_1 \sqcup \ldots \sqcup R_m)^* \bullet ((\neg C_1 \sqcup D_1) \sqcap \ldots \sqcap (\neg C_n \sqcup D_n))$$

onde $R_1,\ldots,R_m$ são os nomes de papéis usados em d ou Σ. Note que esta codificação usa uma descrição de conceito em um dialeto LD decidível em tempo exponencial determinístico.

Como para as regras normais do tableau, as novas regras da Figura 17 são expressas em termos de:

- *assertivas de conceito* da forma $C(a)$, onde C é um conceito atômico e a é uma constante
- *assertiva de papel* da forma $P(a,b)$, onde P é um papel atômico e a e b são constantes
- *assertiva de igualdade* da forma $a = b$, onde a e b são constantes.

Como em [4], assume-se que assertivas de igualdade são *simétricas*, no sentido que, se um nó do tableau contém "$a = b$", então ele implicitamente também contém "$b = a$". Perante as assertivas de igualdade, deve-se também desconsiderar a suposição de unicidade de constantes.

Regra	**Descrição da Regra**
Subconjunto	se "$C(a)$" está no nó N e existe uma restrição de subconjunto $C \sqsubseteq D$ em Σ, então adicionar "$D(a)$" a N
Domínio	se "$P(a,b)$" está no nó N e $\exists P \sqsubseteq D$ é uma restrição de domínio em Σ para a propriedade P então adicionar "$D(a)$" a N
Imagem	se "$P(a,b)$" está no nó N e $\exists P^- \sqsubseteq R$ é uma restrição de imagem em Σ para a propriedade P então adicionar "$R(b)$" a N
Conjunto de constantes	se "$\{c_1,\ldots,c_n\}(a)$" está no nó N então adicionar novos nós filhos $N_1,\ldots,N_n$ de N, onde N_i contém $a = c_i$
Igualdade	se "$a = b$" está no nó N e uma expressão φ também está em N, então adicionar "$\varphi[b/a]$" a N, onde $\varphi\ [b/a]$ denota φ onde cada ocorrência de b é substituída por a
Composição de papéis	se "$(p \circ q)(a,b)$" está no nó N então adicionar "$p(a,c)$" e "$p(c,b)$" a N, onde c é uma constante nova

Figura 17 – Regras adicionais do tableau

41

As regras da Figura 17 podem ser entendidas como segue:

- A *regra de subconjunto* codifica restrições de subconjunto da forma $C \sqsubseteq D$ em Σ. Esta regra indica que, se é declarado que $C(a)$ vale, então $D(a)$ também deve valer, na presença de um subconjunto de restrições $C \sqsubseteq D$ em Σ.
- A *regra de domínio* captura as restrições de domínio da forma $\exists P \sqsubseteq D$ em Σ. Esta regra indica que, se é declarado que $P(a,b)$ vale, então a deve estar no domínio de P, ou seja, $D(a)$ também deve valer.
- A *regra de imagem* captura as restrições de imagem da forma $\exists P^{-} \sqsubseteq R$ em Σ. Esta regra indica que, se é declarado que $P(a,b)$ vale, então b deve estar na imagem de P, ou seja, $R(b)$ deve também valer.
- A *regra de conjunto de constantes* indica que, se é assumido que a está no conjunto de constantes $\{c_1,...,c_n\}$, o que é expresso pela assertiva $\{c_1,...,c_n\}(a)$, então a deve ser uma das constantes, o que é expresso pela assertiva de igualdade $a = c_i$.
- A *regra de composição de papéis* captura composições de papéis da forma $p \circ q$. Esta regra indica que, se é declarado que $(p \circ q)(a,b)$ vale, então deve haver um indivíduo denotado por uma nova constante c tal que $p(a,c)$ e $q(c,b)$ valem.

Pode-se provar que o procedimento do tableau estendido é um procedimento de decisão para o seguinte caso especial do problema de subsunção: “Dado um conjunto Σ de inclusões de um esquema ultra elementar e uma inclusão σ em $\mathcal{ALCVE}$+ [4], determine se $\Sigma \models \sigma$”.

Teorema 1 – Seja $S = (\mathcal{A}, \Sigma)$ um esquema ultra elementar e σ uma inclusão em $\mathcal{ALCVE}^{+}$. Então, qualquer tableau estendido completo para $\Sigma \cup \{\neg\sigma\}$ é fechado sse $\Sigma \models \sigma$.

Teorema 2 – O caso especial do problema de subsunção para esquemas ultra elementares e inclusões em $\mathcal{ALCVE}^{+}$ é decidível.

 42

Para concluir esta seção, apresenta-se um exemplo que ilustra como resolver a instância do problema das restrições de subconjunto do esquema importado descrito no Exemplo 3-3.

Exemplo 3-4 – Resolução das restrições de subconjunto com tableau

Seja o esquema ultra elementar $S = (\mathcal{A}, \Sigma)$ e o esquema importado $I=(\mathcal{K}, \Gamma)$ da Figura 15. Recorde que os conceitos `ai:Product` e `ai:Book` são dois conceitos de K, definidos como (repetindo-se as definições em (7) e (8) abaixo por comodidade):

(7) $\texttt{ai:Book} \equiv \texttt{a:Book} \sqcap \exists\texttt{a:title} \sqcap \exists\texttt{a:pub}$

(8) $\texttt{ai:Product} \equiv \texttt{a:Product} \sqcap \exists\texttt{a:title}$

Para mostrar que $\Sigma \models \texttt{ai:Book} \sqsubseteq \texttt{ai:Product}$ usando tableau, temos que mostrar que $\Sigma \models \texttt{a:Book} \sqcap \exists\texttt{a:title} \sqcap \exists\texttt{a:pub} \sqsubseteq \texttt{a:Product} \sqcap \exists\texttt{a:title}$. Pelo Teorema 1, deve-se construir um tableau, com todos os caminhos fechados, para a expressão

(10) $\Sigma \models (\texttt{a:Book} \sqcap \exists\texttt{a:title} \sqcap \exists\texttt{a:pub}) \sqcap \neg(\texttt{a:Product} \sqcap \exists\texttt{a:title})$

Veja o procedimento na Figura 18.

1. $((\texttt{a:Book} \sqcap \exists\,\texttt{a:title} \sqcap \exists\,\texttt{a:pub}) \sqcap (\neg\texttt{a:Product} \sqcup \forall\texttt{a:title}.\bot)(\texttt{c})$
2. $(\texttt{a:Book} \sqcap \exists\,\texttt{a:title} \sqcap \exists\,\texttt{a:pub})(\texttt{c})$
3. $(\neg\texttt{a:Product} \sqcup \forall\texttt{a:title}.\bot)(\texttt{c})$
4. $\texttt{a:Book}(\texttt{c})$
5. $(\exists\,\texttt{a:title})(\texttt{c})$
6. $(\exists\,\texttt{a:pub})(\texttt{c})$
7. $\texttt{a:title}(\texttt{c,d})$
8. $\texttt{a:pub}(\texttt{c,e})$

Ramo esquerdo:

9. $\neg\texttt{a:Product}(\texttt{c})$
10. $\texttt{a:Product}(\texttt{c})$
11. X

Ramo direito:

12. $(\forall\texttt{a:title}.\bot)(\texttt{c})$
13. $\bot\ (\texttt{d})$
14. X

Figura 18 - Exemplo de um tableau

Com base nas regras mostradas na Figura 17, será explicado aqui resumidamente como o tableau da Figura 18 é construído. Primeiramente, vale lembrar que o tableau é um procedimento de refutação; portanto, a `Linha 1` declara que existe um indivíduo, denotado por c, que é um contra-exemplo para o que se deseja provar. Logo:

- pela `Linha 1`, c pertence a (`a:Book ⊓ ∃a:title ⊓ ∃a:pub`) e também pertence à negação de (`a:Product ⊓ ∃a:title`) de acordo com (10). As negações são então levadas para dentro da expressão o máximo possível. Em particular, `∃a:title` é transformado em `∀a:title.⊥`.
- as `Linhas 2` a `6` resultam de decomposições pela aplicação das regras de tableau. As `Linhas 4` a `6` são conseqüência da `Linha 2` pela aplicação da regra de conjunção (⊓-rule)
- As `Linhas 7` e `8` resultam da aplicação da regra do quantificador existencial (∃-rule) às `Linhas 5` e `6`.
- O tableau então se ramifica em dois nós: as `Linhas 9` e `12` seguem da `Linha 3` (⊔-rule)
    1. A `Linha 10` é conseqüência da `Linha 4` pela aplicação da regra de subconjunto, pois existe uma restrição de subconjunto `a:Book ⊑ a:Product` em Σ. Então, a `Linha 11` indica que as `Linhas 9` e `10` levam a uma contradição.
    2. A `Linha 13` segue da `Linha 12` (∀-rule). A `Linha 14` indica que a `Linha 13` leva a outra contradição, pois `⊥(d)` é sempre falso (já que ⊥ denota o conjunto vazio).
- Portanto, o tableau fecha.

Pelo Teorema 1, vale então que:

(11) $\Sigma \models$ `ai:Book ⊑ ai:Product`

3.3. Um procedimento de decisão rápido

Nesta seção, primeiramente é introduzido o *procedimento estendido de subsunção estrutural* (Figura 19) que incorpora inferência lógica com restrições de domínio, imagem e classe. O procedimento é uma modificação daquele descrito em [21, 34]. É possível mostrar que o procedimento estendido de subsunção estrutural é um procedimento de decisão para o seguinte caso especial do problema de subsunção: “Dado um conjunto Σ de inclusões de um

 44

esquema ultra elementar e uma inclusão σ em $\mathcal{FL}^-$, determine se $\Sigma \models \sigma$". (Recorde que σ pertence a $\mathcal{FL}^-$ sse σ contém apenas descrições de conjunto, interseções, quantificação existencial restrita e restrição de valor).

Note que este é então um procedimento de decisão para os problemas da Seção 3.1, quando considerado que Σ é um conjunto de restrições de um esquema ultra elementar e σ é restrita a $\mathcal{FL}^-$. Note ainda que, na Figura 19, utiliza-se a definição da restrição de imagem como $\perp \sqsubseteq \forall P{\cdot}C$, o que é equivalente à definição original $\exists P^- \sqsubseteq C$, dada na Seção 2.2.

```
IMPLIES(Σ,σ)
input:     um esquema ultra elementar com um conjunto de restrições Σ
           uma inclusão σ que pertence a FL⁻
output:    True,     sse Σ ⊨ σ
           False,    caso contrário
begin
   Se alguma das conjunções do lado esquerdo de σ é o conceito vazio ⊥, então retorne True
   Elimine os parênteses das conjunções do lado esquerdo e direito de σ
   Normalize σ
   Assuma que σ é da forma
   e1 ⊓...⊓ em ⊓∀P1· g1 ⊓...⊓∀Pn· gn ⊑ f1 ⊓...⊓ fr ⊓∀Q1· h1 ⊓...⊓∀Qs· hs
   if  f1 é uma descrição de conjunto
      then   begin  if  e1 não é uma descrição de conjunto que não define um subconjunto de f1
                           then return False        /* False implica que Σ ⊭ e1 ⊑ f1
                        b= 2
             end
      else b=1
   for i=b to r do
      if não existe ej, com j∈[b,m], tal que exista uma caminho em G*(Σ) de ej para fi
         then return False              /* False implica que, para cada j∈[1,m], Σ ⊭ ej ⊑ fi
   for i=1 to s do
   begin
      Seja Ri a imagem de Qi, isto é,  ⊥ ⊑ ∀Qi· Ri está em Σ
      if  IMPLIES( Σ , Ri ⊑ hi )
                        /* True implica que Σ ⊨ Ri ⊑ hi e, então, Σ ⊨ ∀Qi· Ri ⊑ ∀Qi· hi
         then   begin  /* como ⊥ ⊑ ∀Qi· Ri , se tem Σ ⊨ ⊥ ⊑ ∀Qi· hi
                   Substitua ∀Qi· hi por ⊥ em σ          /* pois, Σ ⊨ ⊥ ≡ ∀Qi· hi
                   if  o lado direito de σ torna-se ⊥ ⊓...⊓ ⊥
                       then return True
                end
         else   if  existe j∈[1,n] tal que Pj = Qi
                   then   if ¬IMPLIES( Σ , gj ⊑ hi ) /* False implica que Σ ⊭ gj ⊑ hi
                              then return False /* e, então, Σ ⊭ ∀Qi· gj ⊑ ∀Qi· hi
                   else return False
   end
   return True                    /* True implica que Σ ⊨ σ
end
```

Figura 19 - Procedimento *IMPLIES*

Seja S = $(\mathcal{A}, \Sigma)$ um esquema ultra elementar. O procedimento depende do conceito de *grafo de dependência estendido* $\boldsymbol{G^*}(\Sigma) = (\boldsymbol{N^*}, \boldsymbol{E^*})$ para Σ, definido como segue:

- $\boldsymbol{N^*}$ é o conjunto de conceitos atômicos que ocorrem em Σ, estendido com todas as descrições de conceito da forma $\exists P$ para as quais $\exists P \sqsubseteq D$ é uma restrição de domínio em Σ
- $\boldsymbol{E^*}$ contém um arco (A,B) para cada restrição de subconjunto $A \sqsubseteq B$ em Σ, e um arco $(\exists P,D)$ para cada restrição de domínio $\exists P \sqsubseteq D$ em Σ.

O procedimento também depende da noção de *forma normal*, definida a seguir.

Definição 1 – Uma descrição de conceito f de uma linguagem $\mathcal{FL}^-$ está em *forma normal* sse

i. f é uma descrição de conjunto, um conceito atômico ou uma quantificação existencial restrita, ou
ii. f é da forma $e_1 \sqcap ... \sqcap e_m \sqcap \forall P_1 \cdot g_1 \sqcap ... \sqcap \forall P_n \cdot g_n$ onde:
    - e_1 é uma descrição de conjunto (a única descrição de conjunto admissível em f, se for o caso)
    - para todo $i \in [2,m]$, e_i é um conceito atômico ou uma quantificação existencial restrita
    - para todo $i, j \in [1,n]$, se $i \neq j$ então $P_i \neq P_j$
    - para todo $i \in [1,n]$, g_i está na forma normal.

Definição 2 – Uma inclusão $e \sqsubseteq f$ de uma linguagem $\mathcal{FL}^-$ está em *forma normal* sse e e f estão em forma normal.

Uma inclusão σ pode sempre ser normalizada executando-se os seguintes passos:

1. Substitua, em ambos os lados de σ, todas as descrições de conjunto, se for o caso, por uma única descrição de conjunto que representa todas as suas interseções
2. Substitua em ambos os lados de σ, "$(\forall p \cdot e_1 \sqcap \forall p \cdot e_2)$" por "$\forall p \cdot (e_1 \sqcap e_2)$"

3. Reordene as conjunções, se necessário, movendo para o final as descrições quantificadas universalmente (i.e., as descrições de valor restrito). A inclusão resultante é chamada de *forma normal de* σ.

Proposição 1 – Seja $\mathcal{L}$ uma linguagem em $\mathcal{FL}^{-}$. Seja σ uma inclusão de $\mathcal{L}$ e σ' sua forma normal. Então, σ e σ' são logicamente equivalentes.

Portanto, com base na Proposição 1, sempre é possível assumir que σ está em sua forma normal.

Para ajudar a entender o procedimento *IMPLIES* da Figura 19 serão apresentados dois exemplos

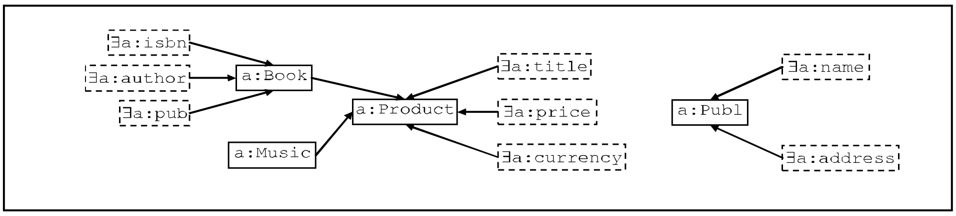

Figura 20 – Grafo de dependência estendido

Exemplo 3-5 – Aplicação do procedimento *IMPLIES* ao Exemplo 3-4

Com base nos esquemas definidos para o Exemplo 3-4, foi construído o grafo de dependência estendido ***G*(Σ)*** da Figura 20. Vale lembrar que `e:Product` e `e:Book` são definidos como:

(1) $\texttt{ai:Product} \equiv \texttt{a:Product} \sqcap \exists\texttt{a:title}$

(2) $\texttt{ai:Book} \equiv \texttt{a:Book} \sqcap \exists\texttt{a:title} \sqcap \exists\texttt{a:pub}$

IMPLIES estabelece que

(3) $\Sigma \models \texttt{a:Book} \sqcap \exists\texttt{a:title} \sqcap \exists\texttt{a:pub} \sqsubseteq \texttt{a:Product} \sqcap \exists\texttt{a:title}$

De fato, no primeiro passo do primeiro laço, *IMPLIES* encontra um caminho de `a:Book` para `a:Product` em ***G*(Σ)***. No segundo passo, *IMPLIES* encontra um caminho trivial, de comprimento 0, de $\exists\texttt{a:title}$ para $\exists\texttt{a:title}$ em ***G*(Σ)***. Então, *IMPLIES* pára, retornando *True*.

Exemplo 3-6 – Problema de inclusão de consulta com *IMPLIES*

Este exemplo ilustra como *IMPLIES* manipula conjuntos de constantes para resolver uma instância do problema de inclusão de consultas. Com os mesmos esquemas usados até aqui, considere as seguintes consultas:

 47

(1) $Q_1 \equiv$ `a:Product` $\sqcap \forall$`a:author.{'Marlowe','Shakespeare'}`

(obter o conjunto de produtos que, se o autor existe, ele é `'Marlowe'` ou `'Shakespeare'`)

(2) $Q_2 \equiv$ `a:Book` $\sqcap \exists$`a:author` $\sqcap \forall$`a:author.{'Shakespeare'}`

(obter o conjunto de livros, com autores conhecidos, cujo autor é `'Shakespeare'`)

Note que Q_1 não é formulada corretamente, pois Σ indica que `a:author` tem domínio `a:Book`. No entanto, *IMPLIES* ainda assim determina corretamente que $\Sigma \models Q_2 \sqsubseteq Q_1$ vale.

O procedimento *IMPLIES* irá fazer uso novamente do grafo de dependências estendido ***G*(Σ)*** mostrado na Figura 20. *IMPLIES* estabelece que $\Sigma \models Q_2 \sqsubseteq Q_1$ como segue.

No primeiro passo do primeiro laço, *IMPLIES* processa `a:Product` e encontra um caminho de `a:Book` para `a:Product` em ***G*(Σ)***. Passando para o segundo laço, no primeiro passo, *IMPLIES* agora processa $\forall$`a:author.{'Marlowe','Shakespeare'}` e executa uma chamada recursiva tendo como parâmetro a inclusão `{'Shakespeare'}` $\sqsubseteq$ `{'Marlowe','Shakespeare'}`. Como o conjunto da esquerda é de fato um subconjunto do conjunto da direita, a chamada recursiva irá retornar *True*. Já que essas são as únicas conjunções de Q_1, *IMPLIES* para, retornando *True*.

O primeiro teorema a seguir estabelece a correção e completude do procedimento *IMPLIES* e o segundo teorema indica que *IMPLIES* é um procedimento polinomial.

Teorema 3 – Seja $\mathcal{S}=(\mathcal{A},\Sigma)$ um esquema ultra elementar e σ um inclusão em $\mathcal{FL}^-$. Então *IMPLIES* (Σ,σ) retorna *True* sse $\Sigma \models \sigma$, caso contrário ele retorna *False*.

A prova deste teorema pode ser encontrada em Lauschner et al [29].

Teorema 4 – O caso especial do problema de subsunção para esquemas ultra elementares e inclusões em $\mathcal{FL}^-$ é decidível em tempo polinomial.

Prova

Seja $\mathcal{S}=(\mathcal{A},\Sigma)$ um esquema ultra elementar e σ uma inclusão em $\mathcal{FL}^-$. Assuma que $\boldsymbol{G^*}(\Sigma)$ possui u vértices e v arestas e que, após a normalização, σ é da forma

$$e_1 \sqcap \ldots \sqcap e_m \sqcap \forall P_1 \cdot g_1 \sqcap \ldots \sqcap \forall P_n \cdot g_n \sqsubseteq f_1 \sqcap \ldots \sqcap f_r \sqcap \forall Q_1 \cdot h_1 \sqcap \ldots \sqcap \forall Q_s \cdot h_s$$

Assuma que o número total de conjunções do lado direito é t, independente do fato das conjunções ocorrerem dentro de restrições de valor ou não. Assim, t é a soma do número de conjunções as quais estão do lado direito da inclusão na chamada inicial, e fora das restrições de valor, mais o número de conjunções as quais estão do lado direito das inclusões passadas como paramento para as chamadas recursivas, e fora das restrições de valor. Para cada uma das conjunções, *IMPLIES* faz ou uma busca de caminho em $\boldsymbol{G^*}(\Sigma)$, ou faz no máximo 2 chamadas recursivas. Consequentemente, incluindo as chamadas recursivas, o corpo do procedimento *IMPLIES* é executado no máximo $2.t$ vezes. Além disso, como $\boldsymbol{G^*}(\Sigma)$ possui u arestas e v arcos, cada busca em $\boldsymbol{G^*}(\Sigma)$ requer $O(Max(u,v))$ passos. Logo *IMPLIES* é $O(t.Max(u,v))$.

3.4.
Conclusões do capítulo

As contribuições deste capítulo são três. Primeiro, mostrou-se como expressar restrições de subconjunto e atribuir definições de domínio e imagem em um dialeto simples de LD, e como reduzir o problema de inclusão de consultas e outros problemas correlatos diretamente ao problema de subsunção. Segundo, foi relatado que através da escolha cuidadosa de um dialeto LD é possível estender o procedimento de decisão de tableau tradicional para o problema de subsunção. Assim, com a introdução de novas regras para o tableau, foi possível levar em conta as classes de restrição de integridade introduzidas e aplicar o procedimento sem fazer uso de (ineficientes) reduções. Terceiro, e mais importante, para as classes de restrições de integridade consideradas modificou-se o procedimento de subsunção estrutural, conservando-o na mesma classe de complexidade.

 49

Tanto o procedimento de tableau quanto o procedimento de subsunção estrutural tem importância prática para derivar as restrições do esquema importado que, como já foi dito, é um dos temas centrais desta tese. De fato, gerar as restrições do esquema importado passa a ser um problema de testar quais restrições são válidas na presença dos axiomas e definições do esquema exportado. Tome, por exemplo, duas restrições C e D do conjunto de restrições do esquema importado. Traduza as restrições para o vocabulário do esquema exportado, desta forma $C \equiv \mu_C$ e $D \equiv \mu_D$. Determinar se $\Sigma \models \mu_C \sqsubseteq \mu_D$, onde Σ é o conjunto de restrições do esquema exportado, é testar se a implicação lógica vale, usando um dos procedimentos propostos neste capítulo. Caso C e D estejam em $\mathcal{FL}^-$ aplica-se o procedimento de subsunção estrutural por ter complexidade polinomial. Caso contrário, aplica-se o tableau. No entanto, os dois procedimentos aqui apresentados ainda precisam ser ajustados para lidar com restrições de cardinalidade, o que será apontado para trabalhos futuros.