UTILIZANDO A CLÁUSULA MAIS ESPECÍFICA E DECLARAÇÃO DE MODOS NA REVISÃO DE TEORIAS DE PRIMEIRA-ORDEM A PARTIR DE EXEMPLOS

Ana Luísa de Cerqueira Leite Duboc

DISSERTAÇÃO SUBMETIDA AO CORPO DOCENTE DA COORDENAÇÃO DOS PROGRAMAS DE PÓS-GRADUAÇÃO DE ENGENHARIA DA UNIVERSIDADE FEDERAL DO RIO DE JANEIRO COMO PARTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE MESTRE EM CIÊNCIAS EM ENGENHARIA DE SISTEMAS E COMPUTAÇÃO.

Aprovada por:

Prof. Gerson Zaverucha, Ph.D.

Prof. Mário Roberto Folhadela Benevides, Ph.D.

Prof. Sheila Regina Murgel Veloso, D.Sc.

Prof. Renata Wassermann, Ph.D.

RIO DE JANEIRO, RJ - BRASIL

MARÇO DE 2008

DUBOC, ANA LUÍSA DE CERQUEIRA LEITE

Utilizando a cláusula mais específica e declaração de modos na revisão de teorias de primeira-ordem a partir de exemplos [Rio de Janeiro] 2008

XIII, 96p. 29,7 cm (COPPE/UFRJ, M.Sc., Engenharia de Sistemas e Computação, 2008)

Dissertação - Universidade Federal do Rio de Janeiro, COPPE

1. Revisão de Teoria
2. Cláusula mais específica
3. ILP
4. Aprendizado de Máquina
5. Lógica de Primeira Ordem

I. COPPE/UFRJ II. Título(série)

Dedicatória

Aos meus pais, sem os quais nada disso seria possível.

"Toda história tem seu fim, mas na vida cada final é um novo começo"

Agradecimentos

A Deus, por estar sempre comigo, por me dar forças para continuar quando pensei em desistir, e por me mostrar que sonhos podem se tornar reais, mesmo que seja difícil realizá-los.

Aos meus pais, Maria Lúcia e Gilson, por sempre me apoiarem e me darem o suporte necessário para que eu chegasse até aqui, por acreditarem em mim, por não duvidarem da minha capacidade, e, o mais importante, pelo amor que me dedicam a cada dia.

À minha irmã, Letícia, pela paciência enorme que teve comigo, principalmente quando nada funcionava no meu computador, e por me incentivar a seguir em frente, querendo sempre o melhor pra mim.

Ao meu orientador, Professor Doutor Gerson Zaverucha, pelos conhecimentos transmitidos, por me guiar na elaboração deste trabalho, por acreditar em mim até quando eu mesma não acreditava mais.

À minha mais nova amiga, Aline Paes, por me ajudar a recomeçar no mestrado, pelos ensinamentos, pela paciência, por tirar minhas dúvidas, pelos conselhos, e por me apoiar sempre.

Aos meus amigos Carina e Alexandre Silva, por sempre me ouvirem, me apoiarem e acreditarem em mim.

Ao meu amigo e professor João Carlos Pereira da Silva, pelas palavras de amizade, por ser meu psicólogo e amigo nos momentos em que mais precisei.

Ao Eric, Rafael e Guilherme, alunos da iniciação científica, pela contribuição feita a este trabalho.

Aos amigos do mestrado e ao restante dos meus amigos e da minha família, por todo apoio, carinho e compreensão.

Ao CNPQ pelo suporte financeiro.

Obrigada a todos que de alguma forma contribuíram para que este trabalho fosse concluído, me apoiando, incentivando e acreditando em mim.

Resumo da Dissertação apresentada à COPPE/UFRJ como parte dos requisitos necessários para a obtenção do grau de Mestre em Ciências (M.Sc.)

UTILIZANDO A CLÁUSULA MAIS ESPECÍFICA E DECLARAÇÃO DE MODOS NA REVISÃO DE TEORIAS DE PRIMEIRA ORDEM A PARTIR DE EXEMPLOS

Ana Luísa de Cerqueira Leite Duboc

Março/2008

Orientador: Gerson Zaverucha

Programa: Engenharia de Sistemas e Computação

Sistemas de revisão de teorias são desenvolvidos para melhorar a acurácia de uma teoria inicial dada, tornando-as mais acuradas e compreensíveis do que as geradas por métodos puramente indutivos. Estes sistemas partem da teoria inicial fornecida e a modifica nos pontos onde há uma falha na classificação dos exemplos. Uma das modificações possíveis é a adição de antecedentes à cláusula sendo revisada. Um conhecido sistema de revisão de teorias é o FORTE, cuja operação de adição de antecedentes é baseada no FOIL, e considera todos os literais da base de conhecimento como possíveis antecedentes a serem adicionados na cláusula. Tal fato acarreta em um espaço de busca muito grande, o que domina o custo do processo de revisão. O estado da arte dos algoritmos de aprendizado em ILP restringem a busca de antecedentes aos literais relevantes para um determinado exemplo positivo. Tais literais compõem a cláusula mais específica, gerada por uma busca direcionada a modos. Visando melhorar a eficiência da operação de adição de antecedentes no FORTE, propomos a utilização da cláusula mais específica como espaço de busca de antecedentes, e a declaração de modos para validar os literais da cláusula mais específica sendo adicionados à cláusula revisada. Resultados experimentais mostram que o processo de revisão passou a ser três ordens de magnitude mais rápido do que o que é feito no FORTE, e teorias mais compreensíveis foram geradas sem prejuízo da acurácia. A abordagem proposta também melhora significativamente a acurácia de teorias geradas a partir do sistema Aleph, o mais utilizado sistema de ILP.

Abstract of Dissertation presented to COPPE/UFRJ as a partial fulfillment of the requirements for the degree of Master of Science (M.Sc.)

USING THE MOST ESPECIFIC CLAUSE AND MODES DECLARATION ON FIRST-ORDER REVISION OF THEORIES FROM EXAMPLES

Ana Luísa de Cerqueira Leite Duboc

March/2008

Advisor: Gerson Zaverucha

Department: Systems Engineering and Computer Science

Theory revision systems are designed to improve the accuracy of an initial theory, producing more accurate and comprehensible theories than purely inductive methods. Such systems start from an initial theory and modify it on points where some example is misclassified. One of the possible modifications is the addition of antecedents to the clause been revised. A known theory revision system is FORTE. The adding antecedents operation of FORTE is based on FOIL, and therefore considere all the literals of background knowledge as possible antecedents to be added to the the clause. This fact implies in a huge search space which dominates the cost of the revision process. The state of the art of learning algorithms in ILP restrict the search of antecedents to literals that are relevant to a positive example. These literals form the most specif clause, which is generated through a mode directed search. Aiming to impove the efficiency of the antecedent adding operation in FORTE, we propose the use of the most specif clause as antecedent search space. Besides that, we also propose the use of modes to define which literals of the most specif clause can effectively be added to the clause been revised. Experimental results show that the runtime of the revision process is on average three orders of magnitude faster than the one that is done on FORTE, and generate more comprehensible theories without decreasing the accuracy. The proposed approach also significantly improves predictive accuracy over theories generated by Aleph system, the most used system in ILP.

Sumário

Lista de Abreviaturas

BC *Bottom Clause*

C Cláusula a ser especializada

E Conjunto de exemplos fornecido ao sistema de revisão

H Teoria de primeira-ordem

ILP *Inductive Logic Programming*

BK Conhecimento preliminar

Lista de Algoritmos

Lista de Figuras

Lista de Tabelas

Capítulo 1

Introdução

A lógica de primeira-ordem é um sistema robusto de representação de conhecimento, pois consegue representar situações complexas envolvendo vários objetos e relações entre eles. Dentro da área de Programação em Lógica Indutiva(ILP) (MUGGLETON, 1991), (MUGGLETON, 1992a), (LAVRAC, DZEROSKI, 1994), (MUGGLETON, De RAEDT, 1994), (MUGGLETON, 1999), definida como a interseção entre o aprendizado indutivo e a programação lógica, algoritmos de aprendizado têm sido desenvolvidos para aprender teorias em lógica de primeira-ordem. A partir de uma teoria inicial assumida como correta e que portanto não pode ser modifcada, o que chamamos de conhecimento preliminar, e de um conjunto de exemplos positivos e negativos, sistemas ILP acham um conjunto de novas cláusulas que implique logicamente os exemplos positivos e não implique os exemplos negativos.

Enquanto o aprendizado indutivo lida com aprender teorias a partir do zero, já que não considera teorias iniciais dadas que fazem parte do conhecimento preliminar, a área de ILP também engloba a tarefa de revisar teorias, onde, além do conhecimento preliminar, partimos de uma teoria inicial que pode esta incorreta ou imcompleta, e portanto pode ser modificada. Portanto o objetivo é achar uma teoria revisada que passe a implicar logicamente os exemplos positivos e não implique logicamente os exemplos negativos.

A única diferença em relação aos sistemas ILP é que uma teoria inicial incorreta e modificável é dada. Os sistemas de revisão de teorias tem como objetivo gerar teorias revisadas mais acuradas do que a inicial.

Sistemas de revisão de teorias partem da teoria inicial fornecida e a modifica

nos pontos onde é verificada uma classificação incorreta dos exemplos. A melhor modificação entre todas as revisões propostas é escolhida. As modificações são feitas utilizando-se operadores de revisão, que se dividem em operadores de especialização e generalização. Pontos na teoria onde a prova de instâncias positivas falha são lugares onde a teoria precisa ser generalizada, e cláusulas usadas em provas de instâncias negativas são pontos onde a teoria precisa ser especializada. Uma das modificações usadas é a adição de antecedentes, que adiciona um antecedente em uma cláusula incorreta para excluir exemplos negativos que estão sendo provados, não permitindo ao máximo possível que exemplos positivos deixem de ser provados.

Existem várias abordagens de revisão de teorias de primeira-ordem(WROBEL, 1996) tais como FORTE(RICHARDS, MOONEY, 1995), AUDREY(WOGULIS, 1991), CLINT(RAEDT, BRUYNOOGHE, 1992), MIS(SHAPIRO, 1983), RUTH(ADé, et al., 1994) e KRT(WROBEL, 1994). De acordo com (WROBEL, 1996), os sistemas GOLEM(MUGGLETON, FENG, 1992) e CIGOL(MUGGLETON, BUNTINE, 1988), apesar de serem sistemas de aprendizado de teorias, podem ser considerados como sistemas de revisões mesmo não aceitando teoria inicial, pois estes sistemas conseguem aprender teorias envolvendo múltiplos predicados. O sistema de revisão de teorias FORTE difere da maioria dos sistemas citados pois faz revisão automática ao invés de interativa com o usuário e utiliza operadores de generalização e especialização como espaços de busca, entre eles o operador de adição de antecedentes.

A Maioria dos sistemas ILP, tal como o Aleph/Progol (SRINIVASAN, 2001), (MUGGLETON, 1995), utilizam o algoritmo de cobertura, onde hipóteses são construídas iterativamente cláusula por cláusula. O algoritmo de cobertura é guloso no sentido de que cada iteração adiciona a melhor cláusula de acordo com algum critério de avaliação local, o que leva à geração de cláusulas localmente ótimas. Além disso, apenas um predicado é aprendido, o que resulta em muitas cláusulas com o mesmo predicado na cabeça e desnecessariamente grandes(BRATKO, 1999). Já o aprendizado de teorias feito pelo FORTE realiza um aprendizado global, considerando cláusulas previamente criadas quando se aprende uma nova cláusula, não usando portanto o algoritmo de cobertura e consequentemente gerando teorias menores.

No FORTE, a busca de antecedentes a serem adicionados pode ser feita através

de dois algoritmos: *hill climbing* e *relational pathfinding*. O *hill climbing* adiciona um antecedente de cada vez, procurando pelo que proporciona melhor desempenho à teoria. O *relational pathfinding*(RICHARDS, MOONEY, 1992) adiciona vários antecedentes de uma só vez, tendo como idéia principal a busca por um caminho num domínio relacional. Ao contrário do *hill climbing*, o *relational pathfinding* é um método de adição de antecedentes desenvolvido para escapar do máximo local. Em ambos os algoritmos a busca por tais antecedentes é uma busca *top-down* baseada no FOIL(QUINLAN, 1990), e o espaço de busca de literais é muito grande pois considera toda a base de conhecimento, sendo que alguns destes literais podem não cobrir nem ao menos um exemplo positivo, o que domina o custo do processo de revisão, pois avaliar cada literal é uma operação muito custosa. A base de conhecimento inclui, entre outras coisas, a definição dos predicados e os tipos de seus argumentos.

O estado da arte dos algoritmos de aprendizado em ILP restringem a busca de antecedentes aos literais relevantes para um determinado exemplo positivo. Tais literais compõem a cláusula mais específica, também chamada de *bottom clause* (MUGGLETON, 1995), que é gerada através de uma busca direcionada a modos. Um sistema conhecido que utiliza a *bottom clause* como espaço de busca para aprendizado de teorias é o Aleph/Progol (SRINIVASAN, 2001), (MUGGLETON, 1995). O Aleph utiliza o algoritmo *hill climbing* para buscar antecedentes, e recentemente foi incluído no Aleph a utilização do algoritmo *relational pathfinding* (ONG, et al., 2005), tendo como objetivo estender o algoritmo *pahtfinding* para fazer uso de declaração de modos, aplicando esta extensão à *bottom clauses*, e passando a buscar caminhos relacionais na cláusula mais específica como forma de restringir o espaço de busca.

Visando melhorar a eficiência da operação de adição de antecedentes no sistema FORTE, propomos a utilização da cláusula mais específica como espaço de busca de antecedentes, tanto na abordagem *hill climbing* quanto no algoritmo *relational pathfinding*. Além disso, propomos também a utilização de declaração de modos para definir quais literais da cláusula mais específica poderão efetivamente ser adicionados à cláusula sendo revisada, pois através dos modos podemos definir se um predicado deve estar na cabeça ou no corpo de uma cláusula, podemos definir

o número máximo de instanciações de um predicado na cláusula e também podemos restringir os argumentos dos predicados a serem de entrada, saída ou constante. Esperamos com isso restringir a busca aos literais mais relevantes, e consequentemente reduzir o tempo de busca e revisão de teorias, mantendo a geração de teorias mais acuradas.

Além disso, sabemos que existem algoritmos de aprendizado de teorias, tal como o Hyper(BRATKO, 1999), que ao contrário do Aleph fazem aprendizado de múltiplos predicados simultaneamente, e que conseguem obter teorias menores e mais acuradas, porém em um tempo muito demorado. Portanto, como um resultado complementar, queremos mostrar que aprender teorias usando técnicas de revisão juntamente com a utilização da cláusula mais específica, é melhor do que o algoritmo de cobertura do Aleph, e que além de conseguir teorias mais acuradas e menores, também consegue um tempo compatível ao do Aleph.

A presente dissertação está organizada da seguinte forma: no capítulo 2 serão descritos conceitos básicos relacionados com a nossa pesquisa. No final do capítulo 2 mostramos alguns trabalhos relacionados a esta dissertação, discutindo outros motivos pelos quais o FORTE foi escolhido como sistema a ser usado na nossa proposta. No capítulo 3 introduzimos a nossa proposta de utilização de modos e da cláusula mais específica para revisar teorias de primeira-ordem a partir de exemplos. Os resultados experimentais são descritos no capítulo 4, e finalmente no capítulo 5 apresentamos nossas conclusões e trabalhos futuros.

Capítulo 2

Conceitos Preliminares

Neste capítulo apresentaremos as técnicas nas quais baseamos nossa pesquisa e que serão importantes para o entendimento do restante da dissertação. Os leitores já familiarizados com algum desses assuntos podem encaminhar-se para as outras seções.

Este trabalho compreende temas como sistemas de representação em lógica de primeira-ordem, aprendizado de teorias utilizando-se a cláusula mais específica e sistemas de revisão de teorias. Portanto, em um primeiro momento serão apresentados conceitos relacionados à lógica e então na seção 2.1 é definida a terminologia utilizada em lógica de primeira-ordem (CASANOVA, et al., 1987), (LLOYD, 1987), (ZAVERUCHA, 2008), na seção 2.2 definimos Programação em Lógica Indutiva(ILP) (MUGGLETON, 1991), (MUGGLETON, 1992a), (LAVRAC, DZEROSKI, 1994), (MUGGLETON, De RAEDT, 1994), (MUGGLETON, 1999), na seção 2.3 descrevemos o PROGOL(MUGGLETON, 1995), (M. FERRO, 2007) que introduz a geração da cláusula mais específica, e na seção 2.4 descrevemos revisão de teorias de primeira-ordem (WROBEL, 1996), dando ênfase ao sistema FORTE(RICHARDS, MOONEY, 1995). Por fim, na seção 2.5 citamos alguns sistemas de revisão de teorias e suas diferenças em relação ao FORTE.

2.1 Lógica de primeira-ordem

A lógica de primeira-ordem, conhecida também como cálculo de predicados, é um sistema lógico que estende a lógica proposicional.

A lógica de primeira-ordem é capaz de representar relações entre objetos, con-

cluir particularizações de uma propriedade geral dos indivíduos de um universo de discurso, assim como derivar generalizações a partir de fatos que valem para um indivíduo arbitrário do universo de discurso. Para ter tal poder de expressão, a linguagem de primeira-ordem usa símbolos mais sofisticados do que os da linguagem proposicional. Para expressar propriedades que valem para todos os indivíduos ou apenas para alguns, são utilizados os quantificadores $\forall$(universal) e $\exists$(existencial), respectivamente.

Daremos a seguir alguns conceitos e terminologias referentes à lógica de primeira-ordem.

Um *alfabeto de primeira-ordem* consiste de:

a) Símbolos lógicos:

a1) Pontuação: "(",",",",")"

a2) Conectivos: $\neg$(negação),$\wedge$(conjunção), $\vee$(disjunção), $\leftarrow$(implicação) e $\leftrightarrow$(bi-condicional)

a3) Quantificadores: $\forall$(universal) e $\exists$(existencial)

a4) Variáveis, que se iniciam com letras maiúsculas. Exemplos: A, Var

a5) Símbolo de igualdade (opcional): $=$

b) Símbolos não-lógicos: funções(que se iniciam por letras minúsculas), constantes(símbolos funcionais de aridade 0) e predicados.

O conjunto de *termos de primeira-ordem* é o conjunto formado pelas variáveis, constantes, e funções aplicadas a termos, como por exemplo $f(t_1, ..., t_n)$, onde $t_1, ..., t_n$ são termos. Se $t_1, ..., t_n$ são termos e P é um símbolo predicativo n-ário, então $P(t_1, ..., t_n)$ é chamado de fórmula atômica ou átomo. Exemplo: *tio(joão,X).* Um átomo é um literal positivo, e a negação de um átomo é um literal negativo.

Se A e B são fórmulas, então ($\neg$A), (A$\wedge$B), (A$\vee$B), (A$\leftarrow$B) e (A$\leftrightarrow$B) são fórmulas. Se B é uma fórmula e X é uma variável, então $\forall_X$(B) e $\exists_X$(B) também são fórmulas.

Uma *cláusula* é uma disjunção de literais precedida por um prefixo de quantificadores universais, um para cada variável que aparece na disjunção, na seguinte forma:

$$\forall X_1 \forall X_2 ... \forall X_s (L_1 \vee L_2 \vee ... \vee L_m)$$

onde cada L_i é um literal e $X_1, X_2, ..., X_s$ são todas as variáveis ocorrendo em $(L_1 \vee L_2 \vee ... \vee L_m)$. O conjunto $\{A_1, A_2, ..., A_h, \neg B_1, \neg B_2, ..., \neg B_b\}$, onde A_i e B_i são átomos, indica a cláusula

$$\forall A_1 \forall A_2 ... \forall A_h \forall B_1 \forall B_2 ... \forall B_b (A_1 \vee ... A_h \vee \neg B_1 \vee ... \vee \neg B_b)$$

que é equivalentemente representado por $\{A_1 \vee ... \vee A_h \leftarrow B_1 \wedge ... \wedge B_b\}$, e pode ser escrita como $A_1, ..., A_h \leftarrow B_1, ..., B_b$, onde $A_1, ..., A_h$ é a cabeça e $B_1, ..., B_b$ é o corpo da cláusula. Normalmente em ILP é usado :- ao invés de $\leftarrow$. Uma *teoria* é um conjunto de cláusulas, representando a conjunção dessas cláusulas.

Uma *cláusula de Horn* é uma cláusula que contém no máximo um literal positivo, ou seja, um literal na cabeça. Cláusulas com exatamente um literal na cabeça são chamadas de *cláusulas definidas*. Um *fato* é uma cláusula definida com o corpo vazio, representado por $A \leftarrow$ e denotado por A. Um conjunto de cláusulas definidas é chamado de *programa lógico definido*. A figura 2.1 mostra um exemplo de um programa lógico definido.

```
pais(jorge,julio).        pais(jorge,lucia).        genero(jorge,masculino).
genero(julio,masculino).  genero(lucia,feminino).

pai(X,Y) :- pais(X,Y), genero(X, masculino).
mae(X,Y) :- pais(X,Y), genero(X,feminino).
irmao(X,Y) :- irmaos(X,Y), genero(X,masculino).
irma(X,Y) :- irmaos(X,Y), genero(X,feminino).
irmaos(X,Y) :- pais(Z,X), pais(Z,Y), X \= Y.
```

Figura 2.1: Exemplo de um programa lógico definido

Uma *substituição* $\theta = \{V_1/t1, ..., V_n/t_n\}$ é uma associação de termos t_i para variáveis V_i. Aplicar uma substituição θ para um termo, átomo ou cláusula F produz o termo, átomo ou cláusula instanciado $F\theta$, onde todas as ocorrências de variáveis V_i são simultaneamente substituídas pelo termo t_i. Um termo, átomo ou cláusula é básica quando não existe nenhuma variável ocorrendo nele. Uma cláusula é livre de funções se ela não contém símbolos funcionais. Seja $\sum = \{E_1, ..., E_n\}$ um

conjunto de fórmulas e β uma substituição. β é um unificador de $\sum$ se, e somente se, $E_1\beta = ... = E_n\beta$.

Uma cláusula C $\theta - subsume(\preceq)$ a cláusula D se existe uma substituição θ tal que $C\theta \subseteq D$. Por exemplo, a cláusula $C : f(A,B) \leftarrow p(B,G), q(G,A)$ $\theta - subsume$ a cláusula $D : f(a,b) \leftarrow p(b,g), q(g,a), t(a,d)$ através da substituição $\{A/a, B/b, G/g\}$. Portanto, $C \preceq D$.

Interpretação (ou atribuição de valor-verdade) é um processo que mapeia uma sentença em alguma declaração em relação a um determinado domínio. Se a interpretação der o valor verdadeiro para uma sentença então ela satisfaz esta sentença e tal interpretação é chamada um *modelo* da sentença. A noção de modelo é aplicada extensionalmente para uma teoria clausal: uma interpretação é um modelo para um conjunto se ela é um modelo para cada um dos membros do conjunto. Uma teoria DB implica logicamente uma cláusula G ($DB \models G$), ou seja, G é *consequência lógica* de DB, se e somente se todo modelo de DB também for modelo de G.

Um *sistema dedutivo SD = (L, AX, R)* é um sistema formado por uma linguagem L, um conjunto de axiomas lógicos AX e um conjunto de regras de inferência R. Uma fórmula G é um teorema (ou G tem uma *dedução* ou *prova* a partir) de um conjunto de fórmulas DB, no sistema dedutivo SD (denotado por $DB \vdash G$), se e somente se, existir uma sequência finita de fórmulas $D_1, ..., D_n$ tal que:

- a) $D_n = G$
- b) Para todo $i \in [1, n]$, uma das condições abaixo é satisfeita:
  - D_i é uma instância de um axioma lógico de AX
  - $D_i \in DB$
  - existem $j, k < i$ tais que D_i pode ser obtido pela aplicação de uma das regras de inferência de R.

2.2 Programação em Lógica Indutiva

Tendo sido definida a terminologia utilizada em lógica de primeira-ordem e em programação lógica, podemos definir brevemente o que vem a ser Programação em Lógica Indutiva (ILP – *Inductive Logic Programming*). Para mais informações sobre o assunto veja (MUGGLETON, 1991), (MUGGLETON, 1992a), (LAVRAC, DZEROSKI, 1994), (MUGGLETON, De RAEDT, 1994), (MUGGLETON, 1999).

Programação em Lógica Indutiva(ILP) é uma área da inteligência artificial que investiga a construção indutiva de teorias de cláusulas de Horn de primeira-ordem a partir de exemplos e de um conhecimento preliminar. ILP é definido como sendo a interseção entre aprendizado indutivo e programação em lógica. Do aprendizado indutivo ILP herda o seu objetivo: construção de ferramentas e técnicas para induzir hipóteses a partir de observações (exemplos) e sintetizar novo conhecimento a partir da experiência. Da Programação em Lógica(LLOYD, 1987), ILP herda o formalismo representacional, como a representação de teorias, hipóteses, exemplos e conhecimento preliminar, técnicas bem estabelecidas e uma profunda base teórica.

ILP pode ser vista como o estudo de métodos de aprendizado para dados e regras que são representados como predicados lógicos de primeira-ordem. Predicados lógicos permitem variáveis quantificadas e relações, e podem representar conceitos que não podem ser expressos usando uma linguagem de descrição proposicional. Uma base de dados relacional pode ser facilmente traduzida em lógica de primeira-ordem e ser usada como fonte de dados para ILP. Como exemplo, considere as seguintes regras escritas na sintaxe Prolog, as quais definem a relação tio/2:

tio(X,Y):- irmao(X,Z), progenitor(Z,Y).
tio(X,Y):- marido(X,Z),irma(Z,W), progenitor(W,Y).

O objetivo de ILP é inferir regras, como as apresentadas acima, dada uma base de dados com fatos e definições lógicas de outras relações. Por exemplo, um sistema de ILP pode aprender as regras para tio/2, chamado de predicado alvo, dado um conjunto de exemplos positivos e negativos das relações tio/2 e um conjunto de fatos

para as relações progenitor/2, irmao/2, irma/2 e marido/2, chamadas de predicados de conhecimento preliminar para os membros de uma família(MOONEY, et al., 2002).

A hipótese fundamental do aprendizado indutivo é que qualquer hipótese descoberta que aproxima bem uma determinada função alvo usando um conjunto suficientemente grande de exemplos de treinamento, também aproximará bem a função alvo sobre outros exemplos não observados (generalização). A partir dos exemplos de treinamento, o sistema de ILP induz um programa lógico correspondente à visão que define a relação alvo, em termos de outras relações que são dadas como conhecimento preliminar. Em sistemas ILP, os exemplos de treinamento, o conhecimento preliminar e as hipóteses induzidas são todos expressos na forma de programas lógicos, sendo que os exemplos de treinamento são representados como fatos da relação alvo a ser aprendida. O conjunto de conhecimento preliminar é definido como extensional se ele é representado como um conjunto de literais básicos, tais como $pai(pedro, ana)$, ou definido como intensional, se é representado como uma teoria de cláusulas de Horn. Por exemplo, $avo(X, Y) \leftarrow pai(X, Z), pai(Z, Y)$ e um conjunto de fatos básicos definindo a relação *pai* é considerado um conhecimento preliminar intensional. Podemos definir a tarefa básica de ILP da seguinte forma (LAVRAC, DZEROSKI, 1994):

Definição 2.1. *Dados:*

- *Um conhecimento preliminar invariante* BK
- *Um conjunto de exemplos positivos* E_+ *e negativos* E_-

Achar:

- *Uma teoria* H *que, junto com o conhecimento preliminar* BK*, implique logicamente todos os exemplos positivos (completo),* $BK \cup H \models E^+$ *e nenhum dos exemplos negativos (consistência),* $\forall e^- \in E^- : BK \cup H \not\models e^-$*, e que esteja de acordo com o critério de qualidade, tal como o de minimalidade do tamanho da teoria.*

Para formalizar esta definição vamos definir o que vem a ser a relação de cobertura em ILP: dados BK, H e E definidos como anteriormente, a hipótese H cobre um exemplo $e \in E$ em relação a BK se $BK \wedge H \models e$

As definições acima requerem que a relação alvo c e a teoria H concordem em todos os exemplos de E. Porém não há garantias de que H irá corresponder a c em exemplos não vistos. A acurácia de classificar exemplos não vistos é o principal critério de sucesso do sistema de aprendizado.

Definição 2.2. *Dada uma teoria de primeira-ordem H, a acurácia de classificação de H é medida como sendo a porcentagem de exemplos corretamente classificados pela teoria.*

A transparência de H denota o quanto ela é compreensível por humanos. Uma medida utilizada é o tamanho da hipótese, expressa pelo total de condições no conjunto de regras que formam H.

Sistemas ILP podem aprender um único conceito ou múltiplos conceitos (predicados). O aprendiz pode tentar aprender um conceito do zero ou pode aceitar uma teoria inicial que pode ser revisada no processo de aprendizado.

2.3 PROGOL e a construção da *bottom clause*

Em ILP, a busca pela hipótese exige uma heurística de busca no espaço de hipóteses. Geralmente existem duas abordagens: top-down e bottom-up. As duas abordagens, top-down e bottom-up, podem ser vistas como um tipo de algoritmo de cobertura (*covering*). Porém elas diferem na forma como a cláusula é construída. Na abordagem top-down(QUINLAN, 1990), a cláusula é construída do mais geral para o específico, sendo que a busca geralmente começa da cláusula mais geral e sucessivamente a especializa com predicados do conhecimento preliminar, de acordo com alguma heurística de busca. Na abordagem bottom-up(MUGGLETON, 1992a), a busca começa com a hipótese mais específica, com o conjunto de exemplos, e constrói as cláusulas do específico para o geral através da generalização de cláusulas mais específicas. A Abordagem top-down mais popular em ILP é o FOIL(QUINLAN, 1990), enquanto que a abordagem bottom-up mais popular em