Joelson Nogueira de Carvalho

S3O: um método de busca de similaridades em objetos estruturados

Campina Grande
1999

Joelson Nogueira de Carvalho

S3O: um método de busca de similaridades em objetos estruturados

Dissertação apresentada ao curso de Mestrado em Informática do Departamento de Sistemas e Computação da Universidade Federal da Paraíba, como requisito parcial para a obtenção do título de *Mestre em Informática.*

Área de concentração: Ciência da computação
Linha de pesquisa: Inteligência Artificial
Orientador: Dr. Edilson Ferneda

UNIVERSIDADE FEDERAL DA PARAÍBA
Campina Grande
1999.

FICHA CATALOGRÁFICA ELABORADA PELA BIBLIOTECA CENTRAL DA UFCG

C331m Carvalho, Joelson Nogueira de
S30 : um metodo de busca de similaridades em objetos estruturados / Joelson Nogueira de Carvalho. - Campina Grande, 1999.
159 f.

Dissertaca (Mestrado em Informatica) - Universidade Federal da Paraiba, Centro de Ciencias e Tecnologia.

1. Inteligencia Artificial 2. Aprendizagem de Maquina 3. Sistema S30 4. Dissertacao - Informatica I. Ferneda, Edilson II. Universidade Federal da Paraiba - Campina Grande (PB)

CDU 004.8(043)

S3O: UM MÉTODO DE BUSCA DE SIMILARIDADES EM OBJETOS ESTRUTURADOS

JOELSON NOGUEIRA DE CARVALHO

DISSERTAÇÃO APROVADA EM 28.07.1999

PROF. EDILSON FERNEDA, Dr.
Orientador

PROF. GIUSEPPE MONGIOVI , D.Sc
Examinador

PROFª MARIA CAROLINA MONARD , Drª
Examinadora

PROF. JOSÉ HAMURABI N. DE MEDEIROS, M.Sc
Examinador

CAMPINA GRANDE – PB

Agradecimentos

Minha mais sincera expressão de agradecimento aos meus amigos e mestres, os professores do DSC, bem como a todos os seus funcionários, alunos e colaboradores, principalmente àqueles que mais diretamente interagi: Edilson Ferneda (meu orientador neste trabalho), Hélio Menezes, José Hamurabi (revisor deste trabalho), Bernardo Lula, Marcos Sampaio, Homero Cavalcanti, Gilseppe Mongiovi, Antão Moura, Marcelo Barros, Agamemnon Lopes, Agenor, Germana Nóbrega, Mário Ernesto, Roberta, Lidiana, Aninha, Manuela, Zeneide, Alberto, Vera, Arnaldo, Ricardo e Everaldo.

Agradeço ainda aos diretores do Hospital Universitário Alcides Carneiro, Drs. Gilvandro, Gilberto e Roberto Siqueira, ao Prof. Ruben Alves e a Janeide Cavalcanti, que colaboraram comigo de várias formas, permitindo inclusive o uso do meu horário de trabalho naquela instituição para as minhas pesquisas.

Finalmente, agradeço a minha família pelo apoio e força alegremente dispensados.

Aos meus pais,
a Lilian, Daniella e Arthur.

Resumo

Apresentamos neste trabalho um sistema de aprendizagem de máquina a partir de objetos formados por exemplos e contra-exemplos do conceito a ser aprendido. Esses objetos são descritos estruturalmente em uma linguagem de descrição baseada em grafos e o conhecimento é encapsulado em uma estrutura que envolve classes de objetos e relações sobre as mesmas. O sistema apresenta ainda um controle heurístico baseado na lógica majoritária e permite a interferência do usuário como meio de melhorar a aprendizagem.

Esse tipo de sistema de aprendizagem pode ser utilizado em várias aplicações. Neste trabalho, em particular, apresentamos o mesmo como ferramenta de base para o sistema SAID, um sistema de apoio à descoberta científica baseado em aprendizagem de máquina.

Abstract

In this work we present a machine learning system which starts from a set of objects formed by positive and negative samples of the concept to be learned. Each of these objects is structurally described through a graph based language and the knowledge is fitted into a frame that involves classes of objects and relationships. The system also presents a majoritary logic based heuristic control and allows the user's interference as a mean for improving the learning.

Such type of learning system can be used in several applications. In this work, in particular, it is presented as a base tool for the system SAID, a scientific discovery support system based on machine learning.

Sumário

Lista de figuras

Lista de tabelas

Introdução

A Inteligência Artificial (IA), é a tecnologia-chave comum a diversas aplicações que representam o estado da arte dos sistemas computacionais atuais. Na última década, a IA saiu da obscuridade dos institutos de pesquisas e universidades para tornar-se parte do cotidiano de milhões de pessoas, provando não apenas que, ao contrário do que muitos pensavam, trata-se de uma tecnologia tecnicamente viável, mas também que esta tecnologia será imprescindível num futuro bastante próximo. Em seu artigo intitulado "Artificial Intelligence - Realizing the Ultimate Promisses of Computing", David L. Waltz [AIM 1997] apresenta algumas das realizações mais impressionantes da IA, citando entre outras coisas, o sistema "Deep Blue", que conseguiu vencer o mestre enxadrista Gary Kasparov, os sistemas classificadores de objetos espaciais da NASA e o sistema de prova de teoremas matemáticos da ANL (Argonne National Labs), que provou uma conjectura de longa duração sobre álgebra, utilizando um método criado pelo sistema. Na mesma revista, um dos mais renomados autores contemporâneos da IA, Tom Mitchel, professor da Universidade de Carnegie Mellon (CMU), descreve no seu artigo "Does Machine Learning Really Works?", o papel da Aprendizagem de Máquina dentro do contexto do mais vasto campo de pesquisa da Ciência da Computação; numa breve digressão pelos sistemas de aprendizagem de máquina existentes e suas aplicações, o autor relata alguns dos principais paradigmas da área, e focaliza sua discussão sobre três novos nichos, sobre os quais, a aprendizagem de máquina atua de maneira essencial: prospecção de dados (Data Mining), aplicações de difícil programação e aplicações de software sob encomenda. Esses artigos nos dão uma idéia da atual dimensão da IA com destaque para a Aprendizagem de Máquina, e da importância atual e futura desta área de conhecimento.

Agentes Inteligentes

Ainda dentro do contexto dos novos paradigmas da IA, surge uma nova abordagem: tomá-la como uma tentativa de construir sistemas de hardware e de software que sejam capazes de existir em seu próprio ambiente; ou seja, construir sistemas que existam em um ambiente que não seja necessariamente o nosso próprio ambiente [RUSSEL 1995]. Uma inteligência mecânica pode, por exemplo, permear e operar apenas sobre um universo formado por trabalhos científicos publicados, dispostos em uma biblioteca ou na Internet, por exemplo. Um ambiente desse tipo, estaria sempre sujeito a mudanças de parâmetros variados e em tais circunstâncias, é praticamente impossível ter uma des-

crição completa conhecida do mesmo. O que podemos esperar é que tal ambiente possua padrões ou regularidades sobre as quais a máquina possa operar e resolver problemas. Uma máquina operando em tais ambientes, teria que desenvolver e representar internamente o seu próprio conhecimento sobre aquele ambiente; teria que descobrir os problemas a resolver e as regularidades que teria de reconhecer. Se o ambiente for construído por uma pessoa, muitos desses problemas e regularidades poderiam então serem apresentados à máquina automaticamente. Mesmo se esta pessoa não for a responsável pela construção do ambiente, ela pode possuir conhecimento suficiente sobre o mesmo para indicar procedimentos para localizar problemas e regularidades relevantes

Esta discussão traz a tona a mais recente abordagem da IA; a abordagem dos Agentes Inteligentes.

Descoberta Científica

Bem mais antiga que a idéia de agentes Inteligentes, a descoberta científica computacional auxiliada pela IA ganha força com os novos desafios da ciência e revela-se, como veremos, numa importante ferramenta de apoio à pesquisa científica, usando métodos de IA; esta abordagem surge da visão de que:

- A ciência é uma atividade de resolução de problemas
- Heurísticas podem ser aplicadas para a resolução desses problemas
- Métodos de I.A. provêem técnicas para a construção de sistemas computacionais.

Os pioneiros nesse trabalho são Bruce Buchanan [BUCHANAN 1982] e Herbert Simon [SIMON 1977]. Buchanan estabeleceu o novo programa de pesquisa:

> "O problema tradicional de achar um método efetivo para formular hipóteses verdadeiras que melhor explicam um fenômeno, foi transformado no problema de achar métodos heurísticos que gerem explicações plausíveis. O problema de fornecer regras para produzir declarações científicas verdadeiras foi substituído pelo problema de achar heurísticas eficientes selecionar os candidatos considerados razoáveis para uma explicação a partir de um conjunto apropriado de possíveis candidatos"

A descoberta como heurística procura em um espaço de busca, habilitar métodos de IA para que sejam aplicados à tarefas de descoberta.

O primeiro sistema especialista, DENDRAL, era um sistema de descoberta científica capaz de formular hipóteses sobre combinações de substâncias químicas a partir de dados da massa espectrográfica [LINDSAY 1980, 1993]. DENDRAL foi seguido pelo sistema Meta-DENDRAL [MITCHEL 1978], que descobria novas regras em análise espectrográfica de massa de maneira a

contornar o problema de obter regras de peritos [BUCHANAN 1978]. Esses sistemas foram construídos para solucionar tarefas científicas de elevado grau de dificuldade e não foram considerados como a implementação de um modelo de cognição humana.

Uma abordagem mais cognitiva foi realizada no desenvolvimento do sistema BACON [LANGLEY 1987], que redescobriu várias leis científicas da física pesquisando padrões em dados numéricos. Esse sistema foi resultado da evolução de trabalhos de busca padrões em sequências [SIMON 1963] para a busca heurística de padrões em dados numéricos em BACON. Esse sistema também propôs a decomposição de dados relacionais para conjeturar propriedades intrínsecas em um ou mais dos objetos engajados nas relações originais. O grupo também investigou a descoberta dirigida a teoria em STAHL [ZYTKOW 1987]. O sistema KEKADA [KULKARI 1988] modelou padrões de raciocínio em algumas descobertas do bioquímico Hans Krebs, obtendo resultados experimentais surpreendentes que ajudam a dispersar o mistério da ocorrência do acaso na descoberta.

O Agente Racional SAID

Um sistema de apoio à descoberta científica é, antes de mais nada, um sistema de aquisição e geração de conhecimento. Ele verifica princípios, tais como a fase de obtenção de dados, a abstração a partir de dados dentro de um modelo conceitual e a particularização desse último. Nesse contexto, encontramos também a distinção entre conhecimentos profundos, aqueles que se justificam teoricamente e que se comunicam nos escritos científicos, e os conhecimentos situacionais, aqueles geralmente chamados especialistas ou empíricos, que intervêm na implementação dos conhecimentos especialistas e para os quais é desejada a evolução [WILIENGA 1990].

Um sistema desse tipo tem por função assistir a produção de conhecimentos tirando partido ao mesmo tempo do conhecimento teórico sobre o domínio e de um conjunto de dados incompletos, imprecisos e passíveis de erro. Em aquisição de conhecimento, o processo da descoberta tem por objetivo associar essas duas formas de conhecimento pois ele trata das formulações dos conhecimentos capazes de progressão e de revisão.

O Agente Racional SAID [BARBOUX 1990; SALLANTIN 1991A, 1991B; FERNEDA 1992A, 1992B; NGUIFO 1993A, 1993B, 1993C] é um sistema inteligente que tem a capacidade de construir seu próprio conhecimento através da interação com um agente humano e dar explicações para as suas decisões. Na concepção do SAID, Sallantin deu ênfase à concepção de um agente inteligente capaz de desenvolver raciocínios de alto nível, empírico e analógico, a partir de repetições dos três mecanismos básicos de inferência: a abdução, a indução e a dedução. A modelagem do componente de controle do processo de aprendizagem no SAID foi baseado nos princípios da Lógica das Provas e

Refutações proposta por Imre Lakatos em sua Lógica da Descoberta na Matemática [LAKATOS 1976], com a intenção de utilizar os princípios de validação do conhecimento científico formulados na Lógica das Conjecturas e Refutações, concebida por K. Popper como uma lógica da descoberta na Ciência [POPPER 1975A, 1975B, 1993]. Daí a sua denominação para agente racional como um sistema popperiano.

O S3O

Apresentamos nesta dissertação, o sistema de aprendizagem estrutural S3O; Trata-se de um sistema que implementa um método de busca de similaridades em conjuntos de objetos de representação estruturada que permite a geração de conjecturas sobre um determinado conceito a ser aprendido a partir desses objetos. Esse trabalho faz parte de um esforço de grupo para a construção do Agente Racional SAID. Trata-se de uma proposta de um método para implementação do mecanismo de inferência abdutiva para esse Agente Racional.

O projeto onde esse trabalho se insere visa a concepção e experimentação de sistemas de apoio à descoberta. Nesse sentido, esforços se fazem:

- na conceitualização e formalização em três eixos principais:
  - aquisição de novos conhecimentos por aprendizagem automática [FERNEDA 1995A];
  - representação do conhecimento pela introdução de noções tais como as de Teorias Semi-Empíricas [SALLANTIN 1991A, 1991B] e de Agentes Racionais [FERNEDA 1995B];
  - controle por diálogo com o usuário pela troca de exemplos e objeções, diálogo esse materializado através de um protocolo de aprendizagem [NÓBREGA 1998],
- na experimentação dessas idéias sobre problemas reais [FERNEDA 1992, 1994].

Este trabalho vai ao encontro do primeiro dos eixos acima citados; nosso objetivo é o estudo e o desenvolvimento de um método de geração, através de aprendizagem automática a partir de objetos estruturados (ou complexos) na presença de ruído, de uma conjectura sobre um conceito de um certo domínio. Essa aprendizagem deverá se dar em duas etapas:

- Busca de similaridades com a detecção de regularidades e
- Construção de um objeto generalizante (hipótese sobre um conceito do domínio).

Objetivos e Motivação desse trabalho

A proposta do S3O tem como vimos, origem nos trabalhos de Sallantin [SALLANTIN 1991A, 1991B], M. Liquiére [LIQUIÈRE 1990, 1992], Ferneda [FERNEDA 1992, 1994, 1995A, 1995B], e Nóbrega [NÓBREGA 1998]. Com base nesses trabalhos, detectamos a necessidade e a possibilidade da construção do sistema do S3O. A realização desse projeto pode ser descrita pela síntese das seguintes tarefas:

- Estudo do Agente Racional SAID, focalizando seus mecanismos de inferência, principalmente o problema da abdução, aqui vista como busca de similaridades.
- Estudo da Teoria Semi-Empírica (TSE), que permeia o sistema de aprendizagem SAID
- Pesquisa para a escolha ou criação de um método de busca de similaridades apropriado, para extrair os fatos similares que compõem os objetos, cujo conceito se deseja aprender
- Adequação do método de aprendizagem escolhido para o ambiente SAID

Além destas etapas, esse trabalho ainda inclui a implementação de um protótipo de programa de aprendizagem, com a finalidade de permitir a avaliação funcional do método.

A motivação para a realização desse trabalho pode ser estabelecida pela conjunção dos seguintes fatores gerais:

- Inserção do projeto em um contexto bastante atual dentro da IA, onde dentre outros conceitos, interagimos com:
  - Agentes Inteligentes e
  - Descoberta Científica
- Envolvimento com o projeto de desenvolvimento do Agente Racional SAID
  - Desenvolvimento de um subsistema fundamental do SAID

Estrutura da dissertação

A estrutura desta dissertação apresenta o seu corpo principal organizado da seguinte forma:

No Capítulo 1, apresenta-se um breve histórico da IA (ou IA), bem como alguns conceitos que permitem situa-la no contexto das Ciências da Computação. Apresenta-se ainda, de forma bem mais detalhada, comentada e enfática, os principais conceitos de Aprendizagem de Máquina (Machine Learning) relacionados com esse trabalho, com o objetivo de poder nos capítulos seguintes,

estabelecer um referencial para situar o presente trabalho de maneira clara e precisa nesta área do conhecimento.

O Capítulo 2 trata mais especificamente da Aprendizagem de Máquina, apresentando-se aspectos da aprendizagem por exemplos utilizando-se objetos estruturados, além de caracterizar alguns métodos clássicos. Ressalta-se os tópicos que serão mais adiante utilizados para a definição do sistema S3O.

O Capítulo 3 descreve em linhas gerais o sistema de aprendizagem SAID, mostrando sua organização funcional, que lhe permite construir seu próprio conhecimento pela iteração com um especialista do domínio. Apresenta-se ainda, os conceitos de TSE - Teoria Semi-Empírica, de Esquema Mental, de Sistema de Crenças, de Estrutura do Aprendiz, e de Mecanismos de Aprendizagem; permitindo o entendimento desse Agente Racional e sua ligação com o sistema S3O. Além disso, este capítulo apresenta uma breve discussão sobre as inter-relações existentes entre os mecanismos de abdução, indução e dedução.

O Capítulo 4 trata do S3O em si, apresentando o princípio de funcionamento do mecanismo abdutivo, os requerimentos exigidos pelo agente racional SAID

O Capítulo 5 apresenta a implementação do Protótipo S3O, apresentando a interface do mesmo e informações para sua utilização.

Finalmente, o Capítulo 6 apresenta uma breve conclusão do trabalho, assim como as perspectivas de trabalhos futuros.

Capítulo 1

Inteligência Artificial e Aprendizagem de Máquina

1.1 Inteligência Artificial

1.1.1 Definição de Inteligência Artificial

Somos suficientemente inteligentes para entender a inteligência? A *Inteligência Artificial* (ou abreviadamente IA) é possivelmente uma tentativa de responder a essa questão. São várias as definições de IA; elas variam de autor para autor, em função da abordagem adotada. Seguem oito definições de alguns autores bem conceituados da área, enumeradas para que possam ser referenciadas mais adiante:

1. "O novo e excitante esforço para fazer o computador pensar ... máquinas dotadas de mente, no sentido completo e literal" [HAUGELAND 1985]

2. "A automação de atividades que nós associamos ao pensamento humano, atividades como tomada de decisões, resolução de problemas, aprendizagem, ..." [BELLMAN 1978]

3. "A arte de criar máquinas que executam funções que requerem inteligência quando realizadas por pessoas" [KURZWEIL 1990]

4. "O estudo de como fazer computadores realizarem coisas que no momento as pessoas realizam melhor" [RICH 1991]

5. "O estudo das faculdades mentais através do uso de modelos computacionais" [CHARNIAK 1985]

6. "O estudo de computações que tornam possível perceber, raciocinar e agir" [WINSTON 1992]

7. "Um campo de estudo que procura explicar e emular conhecimento inteligente em termos de processos computacionais" [SCHALKOFF 1990]

8. "O ramo da Ciência da Computação que se preocupa com a automação do comportamento inteligente" [LUGER 1993]

Nenhuma definição de IA, como as que vimos acima, é universalmente aceita, mas podemos chegar a um conceito eqüidistante para a grande maioria dos escritores: "IA é a área de estudos voltada para a produção de sistemas artificiais que realizam tarefas que, quando realizadas por seres humanos, exigem inteligência". Esta definição é bastante efêmera e esconde a sua abrangência, sua característica interdisciplinar e a enorme complexidade dos problemas presentemente enfrenta-

dos; apesar disso, esta definição indica claramente seu objetivo e a peculiaridade de sua natureza.

1.1.2 Classificação dos Sistemas de IA

Observando as abordagens expressas pelas definições vistas acima, podemos classificá-las e distribuí-las segundo os termos "pensar" e "agir" sobre os modelos "humano" e "racional" nas quatro seguintes categorias [RUSSELL 1995]:

- **Sistemas que pensam como seres humanos** (definições 1 e 2); esses sistemas expressam a abordagem cognitiva e são na sua grande maioria baseados em modelos oriundos dos experimentos psicológicos, estudados pelas Ciências Cognitivas. Tais modelos são construídos com base nas investigações experimentais em seres humanos (e alguns outros animais), e representam uma das mais importantes contribuições da IA, embora a construção desse tipo de sistemas não seja seu objetivo.
- **Sistemas que agem como seres humanos** (definições 3 e 4); essa abordagem é bem representada pelo clássico *Teste de Turing* [TURING 1950], elaborado para identificar comportamento inteligente, onde uma pessoa interroga duas entidades, uma humana e uma máquina, através de uma interface comum. O sucesso viria com a incapacidade do interrogador em decidir qual entidade seria a máquina. A IA não tem se dedicado a tratar desse tipo de abordagem, pois a expressão da inteligência não existe apenas na interação com humanos.
- **Sistemas que pensam racionalmente** (definições 5 e 6); essa é a abordagem lógica; a abordagem do "pensamento correto". Baseia-se na utilização da lógica formal, que fornece uma notação precisa para as coisas que existem no mundo e as relações entre as mesmas. O desenvolvimento dessa abordagem tem encontrado alguns obstáculos devido a restrições computacionais e a dificuldade de tradução do conhecimento informal para expressá-lo em termos formais. Não se pode porém deixar de incluir esta abordagem como uma das mais importantes, devido ao seu poder de representação e de raciocínio.
- **Sistemas que agem racionalmente** (definições 7 e 8); essa é a abordagem dos Agentes Racionais; a abordagem das crenças e dos objetivos. Agir racionalmente significa alcançar um objetivo de acordo com uma crença. Um Agente é algo que percebe e age. Esta abordagem é indiscutivelmente a mais recente da IA e sua importância é tão grande que alguns autores chegam a redefinir a IA como a área dedicada a construção de Agentes Racionais. Uma breve discussão sobre os agentes inteligentes está descrita na seção 1.2.

Ano	Autor	Contribuição
1943	McCullogh e Pitts	Propuseram um modelo de um neurônio artificial (MCP) e demonstraram que qualquer função computável poderia ser implementada por uma rede formada por seus neurônios artificiais.
1949	Hebb	Modificou o modelo MCP com a regra de atualização, para implementar aprendizagem.
1951	Minsky e Edmonds	Construíram o SNARC; o primeiro computador baseado em redes neurais.
1952	Samuel	Inicia a construção de uma série de programa para jogar Xadrez. Os programas de Samuel eram evolutivos e aprenderam a jogar melhor que seu criador.
1953	Turing	Desenvolve um programa para jogar Xadrez.
1956	Newell e Simon	Apresentam o LT (Logic Theorist), o primeiro programa capaz de raciocínio não-numérico.
	McCarthy	Introduz o termo "Inteligência Artificial" durante uma conferência na Universidade de Princeton.
1958	McCarthy	Apresenta um sistema "Advice Taker", considerado o primeiro sistema completo de IA.
		Definiu a linguagem LISP (LISt Processing). Uma linguagem muito flexível, que permite ao usuário alterar a sua sintaxe.
	Friedberg	Apresenta seus "Algoritmos Genéticos" e prova que um programa pode melhorar seu desempenho selecionando pequenas mutações geradas aleatoriamente em seu código.
1959	Gelernter	Apresenta o GTP (Geometry Theorem Prover), um sistema de resolução geral de problemas na área da geometria.
1960	Widrow	Apresenta ADALINES, uma rede neural para aprendizagem baseada nos trabalhos de Hebb.
1961	Minsky	Apresenta o paper "Towards Artificial Intelligence".
1962	Rosenblat	Apresenta seu PERCEPTRON e com ele prova seu "Teorema da Convergência".
1963	Newell e Simon	Apresentam o GPS (General Problem Solver), Seguindo o sucesso do LT.
	Robinson	Apresenta o "Método da Resolução", um algoritmo para prova de teoremas em lógica de primeira-ordem.
	Slagle	Apresenta o SAINT para resolver problemas de integração.
	Winograd e Cowan	Mostra como um grande número de elementos poderia coletivamente representar um conceito individual.
1967	Bobrow	Apresenta o STUDENT, que interpretava e resolvia problemas algébricos expressos em um subconjunto da linguagem Inglesa.
1968	Evans	Apresenta o ANALOGY, que resolvia problemas de analogia descritos nos testes de QI.
	Raphael	Apresenta o SIR (semantic Information Retrieval), que aceitava declarações em um subconjunto da lingua Inglesa.
1969	Bryson e Ho	Apresentam o algoritmo para a rede neural multicamadas com retropropagação.
	Buchanan, Shuterland, Feigenbaum e Lederberg	Apresentam o DENDRAL; um programa que gerava hipóteses explicativas em química orgânica.
1970	Winston	Apresenta sua teoria da aprendizagem estrutural.
	Montage	Formaliza uma teoria de sintaxe e semântica para linguagem natural, que tem servido de base para muitas interfaces de BDs em IA.

Ano	Autor	Contribuição
1971	Ruffman	Apresenta um programa de visão que interpretava figuras do mundo dos blocos.
1972	Winograd	Apresenta o SHRDLU; um sistema de processamento de linguagem natural.
1973	Colmeraner e Roussel	Desenvolvem PROLOG; uma linguagem de programação declarativa.
	Lighthill	Apresenta seu fatídico Lighthill Report, que causou a suspensão de subvenções oficiais para pesquisa em IA pelo governo britânico.
	Woods	Apresenta o LUNAR; um sistema especialista em geologia com processamento de linguagem natural.
1974	Fahlman	Apresenta o sistema PLANNER, especialista em planejamento.
1975	Waltz	Apresenta um programa de visão com propagação de restrições, que também interpretava figuras do mundo dos blocos.
	Minsky	Desenvolve a idéia de FRAMES.
1979	Duda, Gaschnig e Hart	Apresentam PROSPECTOR; um sistema especialista em prospecção mineral baseado em raciocínio probabilístico.
1980	Herman, Hayes-Roth e Reddy	Apresentam o HEARSAY-II; um sistema de reconhecimento e síntese de voz.
	Marr e Minsky	Desenvolve na DEC (digital Equipment Corporation), o XCON; um sistema especialista para configurar computadores VAX.
1981	Japoneses	Anunciam o projeto "Quinta Geração", para desenvolvimento de computadores inteligentes rodando PROLOG.
	Kuhn, Padua	Editam uma importante séria de papers sobre Processamento Paralelo e sistemas evolucionários.
1982	McDermott	Cria na DEC o R1, primeiro programa de IA com sucesso comercial.
	Hopfield	Cria o neurônio artificial que leva seu nome.
1983	Feigenbaum	Inicia em Stanford o projeto HPP (Heuristic Programming Project); desenhando uma metodologia para construção de sistemas especialistas.
1984	Buchanan, Feigenbaum e Shortliffe	Apresentam MYCIN; um sistema especialista para diagnósticos de patologias infecciosas do sangue.
1986	Horowitz e Heckerman	Lançam a idéia dos Sistemas Especialistas Normativos.
	Rumelhart e McClelland	Apresentam seus trabalhos sobre Processamento Paralelo Distribuído.
1988	Pearl	Desenvolve trabalhos sobre raciocínio probabilístico em sistemas inteligentes.
1989	Berliner	Apresenta o HITECH; o primeiro programa inteligente de xadrez a vencer um grande mestre internacional (Arnold Denker).
1990	Newell, Laird e Rosenbloom	Apresentam o SOAR; conhecido como o melhor exemplo de um agente completo.
1992	Schwuttke	Desenvolve o MARVEL; um sistema especialista que controla transmissão de dados com veículos espaciais em tempo real.
1993	Pomerleau	Desenvolve um robô controlado por redes neurais para guiar automóveis.
1994	Zue, Phillips, Seneff, Paul, Polifroni, Goddeau e Brill	Apresentam o PEGASUS; o melhor programa de reconhecimento de voz até hoje desenvolvido.

Tabela 1.1- Fatos históricos importantes da IA

Podemos ainda notar que as categorias acima descritas podem ser distribuídas em duas principais dimensões:

- Raciocínio e Processos mentais (definições 1 e 3)
- Comportamento ou Ação (definições 2 e 4)

As medidas de sucesso para estas definições são expressas em termos de comparações com dois tipos de parâmetros:

- O desempenho humano (definições 1, 2, 3, 4)
- Um conceito ideal de inteligência (definições 5, 6, 7, 8)

Seguindo estas definições e suas tendências, os estudos de IA têm sido dividido em duas diferentes abordagens:

- Foco sobre pensamento/comportamento humano: esta abordagem desenvolve uma ciência empírica, envolvendo hipóteses e confirmações experimentais
- Foco sobre processos racionais: esta abordagem envolve uma combinação de matemática e engenharia.

1.1.3 Breve Digressão Histórica da IA

Especulações sobre a possibilidade de implementar inteligência mecânica foram originalmente iniciadas quase simultaneamente, por várias pessoas, no início da década de 40, entre as quais podemos destacar Alan Turing, John Von Neumann, Norbert Wiener, Warren McCulloch e Walter Pitts. Entretanto, as experimentações práticas de suas pesquisas em IA não foram possíveis antes dos computadores digitais eletrônicos tornarem-se disponíveis no início da década de 50. Os primeiros resultados práticos e progressos mais rápidos nesta área só ocorreram após a construção de linguagens de processamento simbólico, como IPL e LISP, vários anos mais tarde. Na tabela 1.1 é apresentada uma breve digressão histórica da IA em ordem cronológica das pessoas e dos fatos relevantes para a área [RUSSELL 1995; JACKSON 1995; SHIRAI 1984].

Como podemos ver, a IA não é uma área recente se levarmos em conta a idade dos computadores eletrônicos. O termo foi anunciado pela primeira vez em 1956, mas apenas alguns cientistas o utilizavam. Antes disso porém, leitores e telespectadores de antigas obras de ficção científica, como "The Sci-Fi Magazine" [FULTON 1998] e "The Science Fiction Theather" [FULTON 1998], já anteviam algumas de suas aplicações potenciais, algumas destas hoje realizadas. Outras obras mais recentes desse gênero como "Eu, Robô" [ASIMOV 1994], "Star Trek" [RODDENBERRY 1992], "2001:

Uma Odisséia no Espaço [CLARK 1994] e "Star Wars" [LUCAS 1997] nos apresentam com melhor suporte técnico e alguma base científica, apesar de fantasiosa, uma visão do que a IA poderia vir a produzir algum dia.

Hoje a IA não está presente apenas nas pesquisas; sistemas que utilizam técnicas de IA estão disseminados no cotidiano de milhares de pessoas. Os atuais sistemas inteligentes são capazes de realizar, entre outras coisas, as seguintes tarefas:

1.1.4 Aplicações da IA

- Manipular jogos de estratégia (Xadrez, Poker, Damas, etc);
- Aprender a reconhecer padrões visuais, auditivos, eletrônicos, térmicos, mecânicos, etc.;
- Encontrar provas de teoremas matemáticos;
- Resolver certos problemas bem formulados;
- Processar informação em linguagem natural;
- Controlar processos com refinamento progressivo.

1.1.5 Técnicas e disciplinas da IA

As disciplinas de Inteligência Artificial são matérias de estudo bem definidas, cada uma delas abordando um assunto específico que está relacionado com a área de IA. Podemos quase sempre relacionar cada uma destas disciplinas com capacidades humanas isoladas. Por exemplo, a robótica pode ser considerada como uma tentativa de construir máquinas que realizem tarefas capazes de serem realizadas por membros do corpo humano; sistemas de linguagem natural procuram ampliar a capacidade de comunicação de máquinas aproximando-as da capacidade de comunicação humana, e assim por diante. Os conjuntos de instrumentos desenvolvidos para dotar as máquinas de cada uma destas capacidades são denominados *métodos ou técnicas de IA*. Apresenta-se na Fig. 1.1 os principais métodos ou técnicas de IA atualmente abordados.

Mesmo com a enorme quantidade de técnicas de IA disponíveis hoje, a capacidade das máquinas inteligentes (geralmente computadores), independentemente da ação humana ainda é bastante limitada (A IA mal saiu do seu estágio embrionário e os seus sistemas são capazes apenas de manipular inteligência em níveis rudimentares). Mesmo assim, não se pode deixar de antever as infinitas possibilidades existentes para que estas máquinas venham a demonstrar um comportamento inteligente comparável, e em alguns casos superior, ao humano.

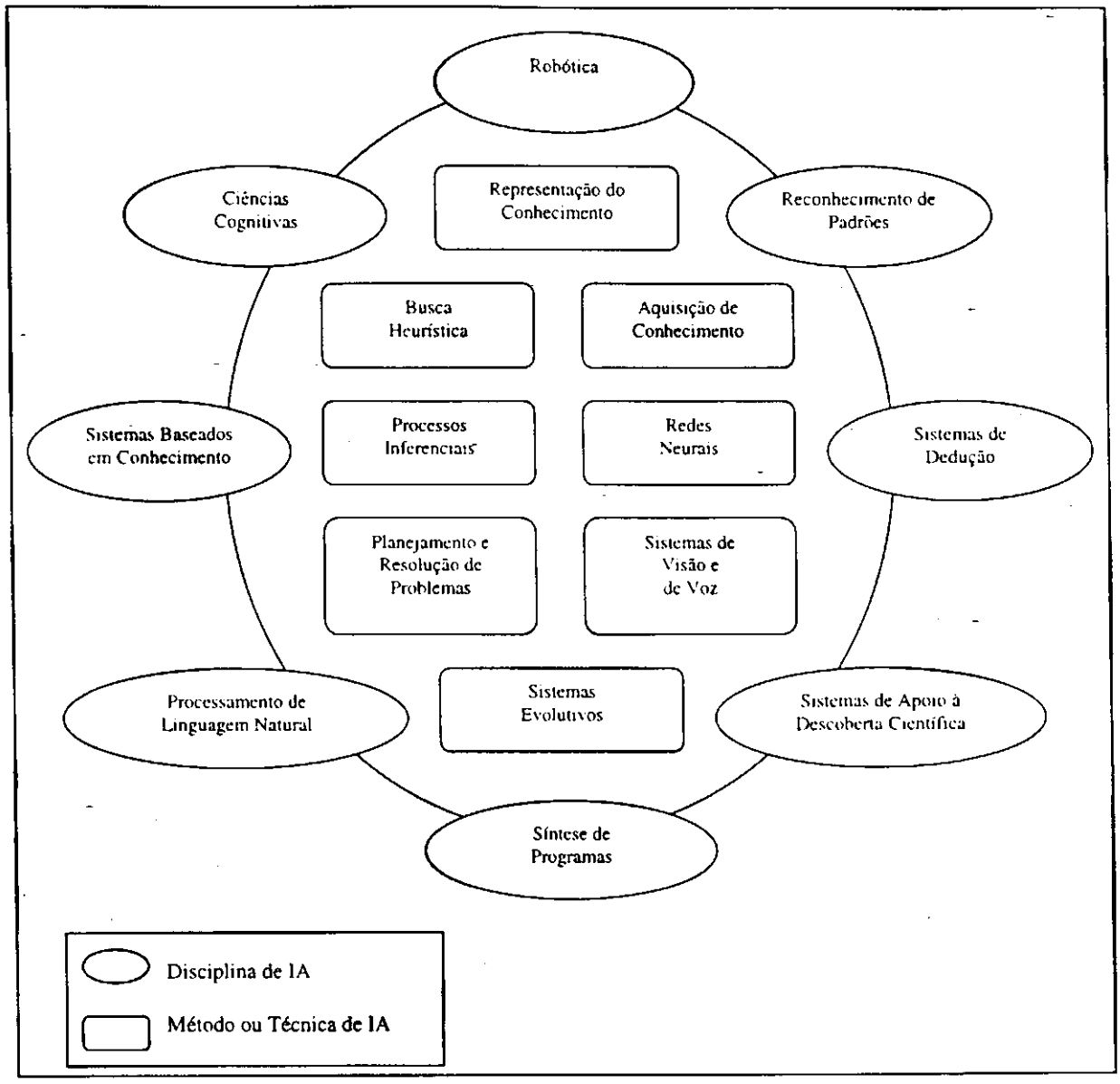


Fig. 1.1 - Métodos e Disciplinas da Inteligência Artificial

1.1.6 Críticas e Limitações da IA

No início, as críticas pareciam ser suficientemente consistentes para destruir qualquer afirmação que fosse relacionada ao tema; isto porque sempre se comparava a produção científica da área às possibilidades humanas (coisa que ainda hoje existe). Uma breve comparação não exaustiva pode ilustrar bem esse fato:

- Semelhanças entre o cérebro humano e um computador:
  - Ambos podem ser considerados dispositivos de uso geral;

- Ambos armazenam e processam informações;
- Ambos utilizam modelos para manipular informações externas ao seu sistema intrínseco.

- Diferenças entre o cérebro humano e um computador:
  - O cérebro possui uma característica de equipotencialidade de sistemas. Em caso de dano de algumas partes do cérebro, outras partes assumem as funções da parte danificada. Em alguns casos, estas funções são reaprendidas, noutros, como no caso dos hábitos, estas funções são obtidas a partir de informações armazenadas dinamicamente em amplas áreas do cérebro. No computador, existe a possibilidade de implantar sistemas redundantes, que poderiam assumir o papel da memória ou do processador, mas estes seriam entretanto predestinados para executar uma tarefa, diferentemente do cérebro, onde as partes que assumem funções de outras partes danificadas não possuem informações antecipadas sobre a nova função.
  - O "dom" é um dote natural inerente aos humanos; uma característica que não parece estar relacionada a qualquer circunstância física. Trata-se da afinidade do ser humano com um ou mais domínios específicos. Esta característica humana encontra ressonância no conceito de "tipo de inteligência": segundo alguns pesquisadores que contestam o conhecido "teste do QI", o referido teste mede apenas um tipo de inteligência, a *inteligência lógica* [GOULD 1983; GARDNER 1983,1993]. segundo eles, existiriam várias outras formas de inteligência, como a musical, a literária, a espacial e a do corpo. Isto explicaria por que algumas pessoas com baixos resultados no QI são, no entanto, extremamente bem-sucedidas. Segundo Gardner, cada um de nós possui, em grau maior ou menor, vários tipos de inteligência; Albert Einstein, Nils Bohr e Stephen Hawkins seriam bons exemplos de inteligência lógica e matemática; Pablo Picasso, Buonarioti Michelangelo e Paul Martin seriam exemplos de inteligência espacial enquanto Wolfgang A. Mozart, Ludwig Van Beethoven e Antonio Vivaldi seriam representantes do grupo que possui inteligência musical superior.
  - A criatividade, no sentido radical da palavra, pode estar estreitamente relacionada com o dom. Ainda não podemos afirmar que esta capacidade poderá um dia ser implementada numa máquina.
  - A velocidade com que as informações circulam é muito maior no computador.
  - A capacidade de armazenamento de informações do cérebro é muito superior à do computador, mas a consistência do armazenamento do computador é superior à do

cérebro. Com isso, o computador pode fazer operações matemáticas muito complexas utilizando uma quantidade mínima de memória.

- O computador é uma máquina passiva, o cérebro é ativo.
- O cérebro é capaz de receber estímulos cujos modelos seriam impossíveis de formalizar matematicamente.
- Estrutura física e funcional das partes são totalmente diferentes.

As limitações tecnológicas existentes antes dos anos 70 também contribuíram para aumentar a descrença. Alguns sistemas de IA que demandam grandes quantidades de memória, alta velocidade e processamento paralelo, como os que incluem percepção de visão e voz, tornaram-se possíveis apenas nos anos finais da década de 80. Porém, o sucesso de alguns dos primeiros trabalhos, proveriam o suporte necessário para que as pesquisas continuassem e fossem intensificadas. Os sistemas dotados de IA eram (e ainda são) construídos para resolver problemas específicos, embora um dos sistemas pioneiros em IA tenha sido concebido como um solucionador de problemas gerais, o GPS de Newell, Shaw e Simon [NEWELL 1963], aplicado a tarefas de senso comum (raciocínio sobre objetos e seus relacionamentos uns com os outros).

É notório que existem limitações para as coisas que um computador pode fazer, para os sistemas que ele pode simular, para os problemas que ele pode resolver e para os procedimentos que ele pode executar. Estas limitações não são apenas relativas aos aspectos físicos das máquinas, como a capacidade de memória e velocidade de processamento (atualmente limitadas na velocidade da luz). Os métodos de IA estão baseados, como todas as ciências, na matemática e em sua capacidade descritiva. Os computadores manipulam dados que são modelos matemáticos de fenômenos do mundo real. A especificação de um determinado objeto pertencente ao mundo real não pode ser feita diretamente no mundo computacional, mas através de uma projeção das características pertencentes a esse objeto sobre elementos pertencentes ao mundo computacional em uma estrutura adequada.

Existem, entretanto, coisas que a matemática não pode descrever completamente, e alguns problemas que podem ser perfeitamente descritos matematicamente podem não ser computáveis, como por exemplo, o problema da parada (Tese de Church) [DAVIS 1983]. Este e outros processos que não podem ser finitamente descritos, não podem ser resolvidos por máquinas dotadas de IA (nem mesmo por humanos). Áreas de conhecimento como Teoria da Computabilidade, Resolução de Problemas e Complexidade de Algoritmos promovem uma abordagem científica a estas questões.

Um outro aspecto importante nesta discussão diz respeito ao grau de adequação dos atuais

computadores digitais ao contexto da IA. É bem verdade que a grande maioria dos produtos advindos da IA até os dias de hoje são baseados na tecnologia digital. Entretanto, alguns autores afirmam com argumentações baseadas estritamente na Física, que o computador digital padrão pode não ser uma ferramenta adequada, pois sua estrutura não é um modelo adequado para um cérebro [BAKER 1997; BRASSARD 1997; BONE 1997; CONVENEY 1995; EKERT 1993; DEUTSCHE 1992]. Um modelo mais adequado seria o *Computador Quântico*; de acordo com os princípios da Física Quântica, um computador quântico poderia ser capaz de simular qualquer sistema físico finito dentro de uma margem de acuidade também finita. Um computador deste tipo teria, por exemplo, a capacidade de gerar números puramente aleatórios [DEUTSCHE 1992].

Vários trabalhos têm sido realizados sobre as limitações teóricas gerais. Paradoxalmente, autores com contribuições significativas para a IA, como Haugeland [HAUGELAND 1981] afirmam a impossibilidade da construção de uma inteligência artificial completa. Outros autores como Searle [SEARLE 1981], Dreyfus [DREYFUS 1981], Davidson [DAVIDSON 1981] e Penrose [PENROSE 1991], chegam através de outras argumentações às mesmas conclusões. Alguns destes autores argumentam que computadores não são capazes de duplicar a bioquímica do cérebro humano, o que os impossibilita de duplicar emoções, sentimentos, etc. Seguindo esse raciocínio, argumentam que "entender" um conceito não pode ser fundamentalmente comparado a uma seqüência de manipulações simbólicas, pois o pensamento humano é "holístico" e não pode ser dividido em sub-processos como se faz nas diversas abordagens de IA. Estas afirmações são contrárias às suposições de alguns outros autores, como McDermott e Woods, que afirmam que o pensamento é realizado através de processos de manipulação simbólica e que o "entendimento" é, essencialmente, um processo que trata "regras conceituais" simbolicamente (estas regras conceituais representariam outros símbolos) [MCDERMOTT 1983; WOODS 1983].

Como vimos, existem inúmeras restrições. Entretanto, nosso conhecimento desses limites não é suficiente para determinar se a IA deve restringir-se às fronteiras determinadas pelas limitações computacionais conhecidas.

1.2 Agentes Inteligentes

1.2.1 Introdução

A abordagem dos Agentes surge como um novo paradigma na Ciência da Computação, na qual os problemas estudados na concepção de sistemas computacionais são organizados sob a ótica de três diferentes perspectivas [WOODRIDGE 1994].

- A Teoria dos Agentes, que se preocupa com a especificação formal de agentes,
- A Arquitetura de Agentes
- As Linguagens Orientadas a Agentes

Essa abordagem computacional evidencia uma nova tendência na concepção de sistemas inteligentes na IA, denominada Agente Inteligente, que pode ser constatada nas declarações seguintes:

> "um dos problemas da IA é descrever e construir um agente inteligente, que perceba o seu ambiente por sensores, tome uma decisão e realize uma ação." [FUKUDA 1997].

1.2.1.1 O que são agentes?

Um Agente é, de maneira geral, qualquer entidade que possua as seguintes características básicas:

- Perceba seu ambiente através de sensores e
- Atue sobre seu ambiente através de mecanismos de ação capazes de modificá-lo.

A estrutura de um agente é como mostra a Fig. 1.2. Tomemos como exemplo o ser humano. Ele possui sensores (olhos, ouvidos, nariz, língua, terminações nervosas na pele - tato) e mecanismos de ação (mãos, pernas, boca, fala, etc...)

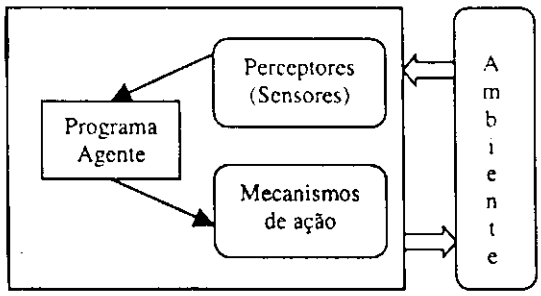


Fig. 1.2 - Componentes básicos de um Agente

1.2.1.2 Agente Racional

A abordagem dos Agentes Inteligentes, segundo Russel e Norvig [RUSSEL 1995], expressa uma integração das diversas técnicas de IA que tratam aspectos isolados da inteligência na concepção de sistemas inteligentes. São várias as concepções de agentes inteligentes existentes atualmente e dentre estas encontra-se a de *Agentes Racionais* - um agente que realiza apenas ações corretas. As ações corretas são aquelas que, quando realizadas, indicam o melhor desempenho do agente na re-

alização das mesmas. Uma medida de desempenho (em função do tempo) nos fornece a informação sobre a atuação do agente e esta medida deve ser imposta por alguém capaz de avaliar esse desempenho. Nada impede que o próprio agente faça sua própria avaliação, mas isto não é uma tarefa trivial. Uma boa opção para avaliar o desempenho do agente é nomear uma entidade externa ao agente para fazê-lo. A necessidade de um componente humano que interaja com os Agentes Racionais pode ser bem expressada pela seguinte definição:

> "Agentes Racionais são sistemas que dispõem de métodos para produzir e controlar seus próprios conhecimentos na interação com um especialista do domínio" [SALLANTIN 1989].

Uma distinção deve ser feita entre racionalidade e onisciência. Um agente onisciente tem o domínio de todo o conhecimento envolvido em suas ações, inclusive o resultado destas ações. Nesse caso, ele poderia autoavaliar-se. Mas a onisciência é impossível de ser alcançada na realidade. Analisando esta última afirmação, como se pode avaliar um agente que em tese deve executar apenas ações que sejam realmente corretas? Se especificássemos um agente levando em consideração essas exigências, seria praticamente impossível construí-lo. Diferente da onisciência, a racionalidade leva em conta o sucesso esperado de acordo com o que foi percebido. Isto permite uma decisão sobre o desempenho com base em comparações entre as opções de ações disponíveis tendo em vista o resultado esperado. A racionalidade é função de quatro variantes:

- A medida de desempenho que mede o grau de sucesso das ações realizadas pelo agente;
- O conjunto de circunstâncias percebidas pelo agente (sequência de percepção);
- O conhecimento do ambiente pelo agente e
- O conjunto de ações disponíveis.

Com base nestas informações, é possível agora definir um Agente Racional Ideal: É um agente que, para cada sequência percebida possível, escolhe dentre o conjunto de ações disponíveis, aquela que, segundo é esperado, maximize sua medida de desempenho, de acordo com as evidências fornecidas pela seqüência percebida e pelo conhecimento do ambiente que o agente possui [NORVIG 1995].

1.2.1.3 Agentes Inteligentes e Busca de Informações

Alguns trabalhos demonstram que os agentes inteligentes podem ser considerados como uma ferramenta para análise de sistemas, utilizados na busca de informações sobre o ambiente [KNOBLOCK 1994A,1994B; LEVY 1994]. Eles preconizam que tais agentes deverão ter as seguintes características:

- Os agentes não serão donos dos dados; somente terão as descrições das fontes de informações e não necessariamente saberão por completo qual o conteúdo dessas fontes.
- Os agentes servirão de mediadores para o processamento do pedido de informações, possivelmente processando dados intermediários (melhor maneira de fazer perguntas, quais fontes de informações pesquisar, etc.)
- Os agentes remeterão a pesquisa pronta para a fonte de informação adequada. Desta forma fica fácil perceber que esses "agentes" podem ser as ferramentas adequadas para a busca de informações que sejam pertinentes aos usuários num curto espaço de tempo, permitindo aos usuários dedicar o seu tempo a outras atividades mais relevantes em suas carreiras.

1.2.2 Agentes, Programas e Arquitetura

Agentes Inteligentes são programas de computadores e/ou hardware que executam taferas destinadas aos seus usuários. A arquitetura de um agente é definida por aspectos do ambiente em que o sistema computável é executado. O termo Agente pode ser considerado como uma composição *programa + arquitetura*. Tipicamente, os Agentes Inteligentes possuem um comportamento orientado a metas numa plataforma computacional heterogênea [HEDBERG 1995]. O que os diferenciam dos softwares e hardwares comuns são as seguintes características [WOOLDRIDGE 1995; COHEN 1994; ETZIONI 1994; KNOBLOCK 1994]:

- **Autonomia:** Um agente tem iniciativa própria, autonomia e tem a capacidade de se planejar para executar determinada tarefa que lhe for proposta, da melhor maneira possível.
- **Prontidão:** Um agente está em constante execução, ou seja, está sempre alerta às funções que deve executar; não a executa somente quando é requisitado.
- **Personalidade:** Um agente possui personalidade confiável e interação amigável com o usuário.
- **Habilidades sociais:** Um agentes interage com outros agentes e usuários através de algum tipo de linguagem de comunicação comum.
- **Adaptabilidade:** Um agente adapta-se às mudanças do meio em que está inserido.
- **Mobilidade:** um agente tem livre acesso a outras máquinas e plataformas.

Um agente não necessariamente deve possuir todas essas características. Esse é um modelo ideal proposto [ETZIONI 1995].

O termo "Agente Inteligente" pode ser encontrado em muitas aplicações sob outro nome, como por exemplo:

- **Interfaces Inteligentes:** agentes inteligentes orientados para comunicação [KAUTZ 1994];
- **Interfaces Adaptativas:** agentes que interligam dois ou mais sistemas comunicantes entre si, adaptando-se a cada nova situação encontrada em cada um deles [KAUTZ 1994];
- **Knowbots:** agentes móveis construídos para serem utilizados em sistemas largamente distribuídos, como a Internet [KAUTZ 1994];
- **Softbots:** agentes construtores de programas [COEN 1994; GENESERETH 1994].
- **Userbots:** agentes construídos para auxiliar usuários em sistemas tipo "helpdesk" [GREIF 1994];
- **Taskbots:** agentes construídos para tratar de tarefas específicas [KAUTZ 1994];
- **Personal Agents:** Taskbots para tarefas personalizadas [GREIF 1994];
- **Network Agents:** Knowbots utilizados em redes de computadores [KAUTZ 1994].

Os Agentes Inteligentes podem constituir uma motivação a mais para os usuários de software, uma vez que: *"A idéia básica de Agentes Inteligentes é a construção de software que auxilie todos os tipos de usuários"* [RIECKEN 1994]. Eles podem ainda configurar um avanço na área de interface homem-máquina. Além disso, também consistirão em um avanço na interface inter-máquina através da construção de colônias de agentes cooperantes entre si.

1.3 Aprendizagem

Seria a aprendizagem uma faceta exclusivamente humana, cuja realização só é possível através do uso da inteligência? Seria a inteligência uma característica exclusivamente humana? Para um índio Sioux, a resposta seria negativa; os Sioux consideravam o *castor* como o animal mais inteligente da criação, pois era capaz de algumas realizações intrigantes como represar um rio, evitar com muita astúcia seus predadores, migrar para um território diferente e em pouco tempo adaptar-se ao novo ambiente. Os Sioux admiravam esse animal pois eles próprios não conseguiam realizar muitas destas façanhas [CURTIS 1975]. Outros como o golfinho exprimem um comportamento inteligente, bastante superior às aptidões naturais necessárias à sua sobrevivência (instinto). Experiências demonstram que alguns animais são realmente capazes de aprender [PASCOE 1998; VAUCLAIR 1987], embora exista uma grande discussão em torno de considerar alguns exercícios de "condicionamento" ao qual alguns animais são submetidos, como uma técnica de ensino. Em qualquer caso, o fato de um animal realizar uma tarefa idealizada por um humano, caracteriza uma aprendizagem. É

bem verdade que aprendizagem é uma característica essencial dos seres inteligentes, cuja definição rigorosa encontra as mesmas dificuldades que a tentativa de definir inteligência. Porém, não se pode afirmar com certeza que se trata de uma aptidão exclusiva dos seres humanos.

A habilidade de aprender é talvez o principal atributo do comportamento inteligente. Trata-se de um fenômeno de facetas múltiplas, realizado através de vários processos onde se incluem:

- aquisição de novos conhecimentos;
- desenvolvimento de técnicas motoras e cognitivas através de instrução ou prática;
- generalização de novos conhecimentos adquiridos;
- representação efetiva de conhecimento;
- descoberta de novos fatos / teorias via observação e experimentação.

Como conseqüência disto, o desenvolvimento de teorias e modelos de sistemas de aprendizagem são de grande importância para os campos de estudos voltados para o entendimento de inteligência, dentre esses a Inteligência Artificial, as Ciências Cognitivas, a Ciência da Informação, a Psicologia, a Epistemologia, a Educação e a Filosofia.

1.4 Aprendizagem de Máquina

1.4.1 O que é Aprendizagem de Máquina?

As pesquisas em torno da aprendizagem realizada por sistemas de computadores (Aprendizagem de Máquina) iniciaram-se na mesma época em que surgiu o termo Inteligência Artificial. Não se trata de uma coincidência, pois existe uma afinidade natural entre as definições de inteligência e de aprendizagem. Estas pesquisas eram baseadas principalmente em modelos construídos a partir da teoria comportamental humana, disciplina da Psicologia que contava com alguns bons trabalhos já naquela época. O que se procurava na verdade, era estabelecer modelos lógicos e matemáticos do raciocínio humano e então utilizá-los em um computador. Os frutos produzidos por estas pesquisas até os dias de hoje formam uma base consistente, porém bastante superficial, para o entendimento dos mecanismos utilizados para a realização desta atividade. Os objetivos das pesquisas de aprendizagem em sistemas artificiais são principalmente [MICHALSKI 1983]:

- a eliminação do elemento humano no processo de aquisição de conhecimento,
- o desenvolvimento de teorias gerais de aprendizagem e
- o ensino assistido por computador.

Uma pesquisa no dicionário nos traz a seguinte definição: "aprendizagem é o meio de adquirir novos conhecimentos". Alguns autores incluem ainda nesta definição: "Meio de reestruturar conhecimentos anteriores" e "Melhoria do desempenho das tarefas". Embora estas duas últimas definições sejam sobrepostas pela primeira, a grande maioria dos autores com interesse em Aprendizagem de Máquina evoca a melhoria de desempenho das tarefas após sucessivas realizações como a que melhor define aprendizagem. Podemos sugerir uma outra definição que parece exprimir melhor esta idéia [SIMON 1983]:

> "A aprendizagem denota mudanças no sistema que são adaptativas no sentido de que elas habilitam o sistema a realizar a mesma tarefa ou tarefas delineadas pela mesma população mais eficientemente a cada vez que são realizadas."

Evidentemente, não é uma tarefa simples aferir correta e completamente a melhoria do desempenho em todos os casos. Uma crítica interessante pode ser feita sobre esta definição; nem sempre existirão mudanças nos resultados dos sistemas que aprendem; podem existir ocasiões em que não seja interessante que um sistema aprendiz melhore seus conhecimentos acima de um determinado patamar. Em outro extremo teríamos os sistemas que receberiam as instruções em um nível tão alto que eles não conseguiriam evoluir acima daquele ponto (saturação de conhecimento). Considerando estas afirmações, poderíamos ter um sistema inteligente que não aprende e apenas aplica seu conhecimento?

Um aspecto interessante dos sistemas de aprendizagem é a existência dos objetivos da aprendizagem: por que aprender determinado conceito? Existem ocasiões em que os objetivos da aprendizagem não são conhecidos por um observador externo. Em outras ocasiões, podem referenciar conceitos vagos e subjetivos, com alto nível de abstração, como por exemplo, apreciar a beleza de um objeto. Procurando um meio de generalizar o conceito para contornar esta complexidade, surge-nos um novo conceito, relacionado com a característica de "utilidade" [MINSKY 1985]:

> "Aprender é fazer mudanças úteis nos trabalhos da nossa mente."

Obviamente, a característica operacional desta definição foi perdida! Como fazer para medir o nível de utilidade de mudanças que otimizam os processos mentais, se esses processos não são exatamente conhecidos? Sabe-se, inclusive, que os processos mentais diferem de pessoa para pessoa. Uma definição mais concreta de aprendizagem, sem recorrer à noção de melhoria de desempenho ou utilidade, parte do princípio de que para adquirir conhecimento sobre qualquer coisa, torna-se necessário que esse conhecimento esteja representado de alguma forma, seja em declarações, procedimentos ou uma mistura dos dois [MICHALSKI 1986]:

> "Aprender é construir e modificar representações do que está sendo experimentado."

Podemos notar que esse último conceito de experiência inclui estímulos sensoriais, bem como os processos mentais internos. Esses dispositivos são o veículo através do qual o sistema de aprendizagem percebe a realidade que está tentando representar. Esse ponto de vista é bastante interessante, pois muda o foco sobre o aspecto central da aprendizagem, que não é mais o de melhorar o desempenho, mas o processo de *construir a representação de alguma realidade*. Evidentemente, a melhoria do desempenho passa a ser uma conseqüência da aprendizagem.

Podemos também notar que essa melhoria poderia ser em alguns casos o objetivo da construção de uma representação. Isto evidencia-se pela afirmação de que *a melhoria do desempenho não é uma condição invariável nos sistemas de aprendizagem*. Um exemplo claro é o caso já citado de um sistema de aprendizagem com alto nível de abstração, como um sistema para apreciar a beleza de um objeto. Em outras situações, pode ser bastante difícil perceber o objetivo da aprendizagem. Por exemplo: um determinado sistema aumenta sua eficiência com o passar do tempo, mantendo sua produção constante (objetivo) e reduzindo o trabalho de maneira inversamente proporcional à aprendizagem. Para um observador externo, o sistema não está aprendendo, pois o objetivo de manter a produção constante não lhe era conhecida e ele baseia sua afirmação na observação da produção que permanece constante, sem evolução.

Podemos dizer, para resumir, que uma máquina aprende sempre que modifica a sua própria estrutura, o programa ou os dados de tal maneira que o desempenho futuro esperado da máquina melhore. Algumas dessas modificações, como a adição de um registro em uma base de dados, não são necessariamente um aprendizado. Mas, por exemplo, quando uma máquina de reconhecimento de voz progride depois de ouvir várias amostras da fala de uma pessoa, trata-se de um caso em que a máquina está aprendendo.

1.4.2 Importância da Aprendizagem de Máquina

Podemos enumerar diversas razões que ressaltam a importância das pesquisas em Aprendizagem de Máquina, entre elas:

- Há tarefas que só podem ser definidas através de exemplos, isto é, somente é possível especificar entradas e saídas, não sendo definida a relação entre elas.
- É possível existir em relações e correlações escondidas em uma grande quantidade de dados. Os métodos de aprendizado de máquina podem ser usados para extrair essas relações; essa abordagem ganha hoje o corpo de uma disciplina de estudos: *Data Mining*.
- Projetistas freqüentemente produzem máquinas que não funcionam como desejado no ambiente de trabalho. De fato, certas características do ambiente não poderiam ser com-

pletamente conhecidas em tempo de projeto. Os métodos de aprendizado de máquina podem ser usados para projetos de máquina que evoluam progressivamente no trabalho.

- A quantidade de conhecimento avaliado em certas tarefas pode ser muito grande para ser codificado explicitamente. Máquinas que aprendem gradualmente podem ser capazes de manipular conhecimento em grandes quantidades com mais eficiência que os humanos.
- Alguns fenômenos que se deseja aprender acontecem em situações que se torna difícil seu acompanhamento por humanos (ambientes como o espaço, grandes profundidades abissais ou o interior de vulcões, observações em períodos de tempos críticos, etc). Uma máquina que aprende pode substituir o agente humano em situações deste tipo.

1.4.3 Como as máquinas aprendem?

Existem muitos fatores que caracterizam os sistemas de aprendizagem de máquina e a maneira pela qual aprendem, mas geralmente, o processo de aprendizagem segue as seguintes fases, de acordo com o esquema mostrado na figura 1.3 [RICH 1997]:

- **Treinamento**: Nesta fase, a informação sobre o que deve ser aprendido é de alguma forma inserida na máquina sob uma determinada representação. A representação do conhecimento aprendido é geralmente armazenada sob a forma de regras.
- **Validação ou Crítica**: Nesta fase, as regras são conferidas e, em caso de necessidade, um novo treinamento é realizado. A validação pode ser realizada com o auxílio de um especialista humano ou com o uso de outro componente baseado em conhecimento. É nesta fase que se verifica a qualidade da aprendizagem.
- **Aplicação**: As regras são usadas em resposta a uma nova situação.

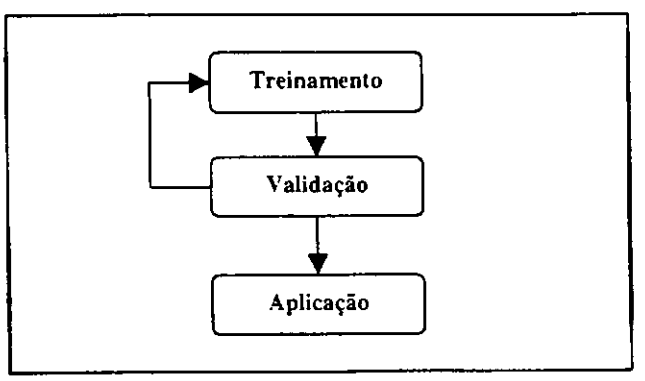


Fig. 1.3 - Passos da Aprendizagem de Máquina

Nem sempre é possível distinguir estas fases. Alguns sistemas de aprendizagem possuem

mecanismos para aprender de maneira que o passo da crítica seja necessário. Noutras situações, o conhecimento sendo aprendido não é passível de crítica, ou não se sabe antecipadamente como fazê-lo. Em alguns sistemas o treinamento não é necessário.

1.4.4 Teoria Formal da Aprendizagem

Pode-se dizer informalmente que em Aprendizagem de Máquina, um aprendiz é qualquer sistema que realiza o mapeamento de um conjunto de evidências em um conjunto de teorias. Esse mapeamento é realizado por uma função. Podemos ainda dizer que o sucesso desses sistemas é obtido sempre que as teorias corretas são escolhidas como resultado desse mapeamento (isto é uma conseqüência direta da escolha da função de mapeamento). Nesse contexto, uma teoria é uma maneira efetiva de enumerar um conjunto de objetos, uma evidência é qualquer objeto finitamente especificável e uma função de mapeamento é um algoritmo de aprendizagem. Os algoritmos de aprendizagem devem explicitar os critérios para o sucesso da aprendizagem e dependem da definição do ambiente.

O formalismo introduzido por Valiant [VALIANT 1984] descreve a *Teoria do Aprendível* que permite estabelecer uma classificação para problemas em função do nível de dificuldade de aprendizagem dos mesmos.

Formalmente, segundo a Teoria do Aprendível, uma máquina aprenderá um conceito se puder, a partir de exemplos positivos e negativos de objetos deste conceito, produzir um algoritmo que classifique exemplos posteriores com certa probabilidade. A complexidade da aprendizagem de um conceito é função de três fatores [RICH 1994]:

- A tolerância a erros.
- O número de características binárias presentes nos exemplos.
- O tamanho da regra necessária para fazer a discriminação.

Se o número de exemplos de treinamento exigidos for polinomial, então o conceito é aprendível.

1.4.5 Pesquisas em Aprendizagem de Máquina

A área de Aprendizagem de Máquina pode ser organizada sob três focos primários de pesquisa [MICHALSKI 1983]:

- Estudos Orientados a Tarefas: desenvolvimento e análise dos sistemas de aprendizagem, visando aumentar a performance sobre um conjuntos pré-determinados de tarefas.

- Simulação Cognitiva: a investigação e simulação em computador do processo de aprendizagem humano.
- Análise Teórica: exploração teórica do espaço de métodos de aprendizagem e algoritmos possíveis, independente do domínio da aplicação.

Devido ao estreito relacionamento entre estas áreas, os progressos obtidos em uma delas é quase sempre acompanhado pelas outras. Isto talvez porque convergem para um ponto em comum, que é a construção de sistemas de aprendizagem em computador.

1.4.6 Paradigmas da pesquisa em aprendizagem de máquina

As pesquisas em Aprendizagem de Máquina abrangem diversas áreas e esforços têm sido aplicados com mais ênfase em algumas áreas que em outras (geralmente, um determinado grupo de pesquisadores está interessado em resolver algum tipo de problema diferente dos outros grupos). Isto acarretou um importante crescimento na diversidade de abordagens do tema aprendizagem em diferentes épocas. Langley [LANGLEY 1996] identifica os cinco principais paradigmas trazidos pelas pesquisas nesta área, que diferem entre si em função da representação da informação e dos algoritmos de aprendizagem.

- **Conexionista**: Utiliza como unidade básica modelos matemáticos de células nervosas humanas e representam o conhecimento como uma rede multi-camada destas unidades limiares que transmitem, a cada ativação, informação dos nodos de entrada, pelas conexões internas até os nodos de saída. Pesos atribuídos em cada conexão permitem controlar o fluxo de informação em cada caso e o ajuste desses pesos incrementam a precisão da informação nos nodos de saída, que podem representar, por exemplo, uma classificação discreta da informação nos nodos de entrada. Redes Neurais são orientadas para a construção de sistemas de aprendizagem de propósitos gerais ou sistemas auto organizáveis.
- **Aprendizagem Baseada em Casos**: O conhecimento é representado em termos de casos específicos. Métodos comparativos recuperam casos para aplicá-los a novas situações. As principais características desta abordagem são o esquema de indexação, a similaridade métrica usada na comparação que recupera os casos relevantes e por fim, no mecanismo que transforma ou adapta os casos encontrados à nova situação.
- **Genética**: A representação da informação é feita em termos de conjuntos de características Booleanas ou binárias (Algoritmos Genéticos) que podem ser usadas como regras de condição ou ação. Utilizam geralmente um processo comparativo lógico, que utiliza pesos (ou potências) atribuídos às regras. De maneira geral, é gerada inicialmente uma popula-

ção formada por um conjunto aleatório de indivíduos que podem ser vistos como possíveis soluções do problema. Durante o processo evolutivo, esta população é avaliada: para cada indivíduo é dada uma nota, ou índice, refletindo sua habilidade de adaptação a determinado ambiente. Uma porcentagem dos mais adaptados são mantidos, enquanto os outros são descartados (darwinismo). Os membros mantidos pela seleção podem sofrer modificações em suas características fundamentais através de mutações e cruzamento (crossover) ou recombinação genética gerando descendentes para a próxima geração. Esse processo, chamado de reprodução, é repetido até que uma solução satisfatória seja encontrada. Embora possam parecer simplistas do ponto de vista biológico, esses algoritmos são suficientemente complexos para fornecer mecanismos de busca adaptativo poderosos e robustos [CARVALHO 1998].

- **Indutiva**: Emprega geralmente regras condição-ação, árvores de decisão ou outra estrutura de conhecimento lógica similar. Geralmente, utiliza-se um processo de comparação lógico para encontrar na estrutura uma regra que case com uma instância. A informação encontrada (classes ou predições) são armazenadas nas folhas da árvore. Os atributos são então avaliados por um algoritmo de aprendizagem que utiliza uma função de avaliação estatística, que os avalia para incorporá-los na estrutura de conhecimento.

- **Analítica**: O conhecimento é geralmente representado em regras em forma lógica. Técnicas de busca são empregadas para resolver problemas. Uma técnica comum nesta abordagem é a representação do conhecimento em termos de regras de inferência, onde se estabelecem problemas como teoremas e utilizam-se buscas para provas. Esses métodos utilizam o conhecimento anterior para construir explicações ou provas, e a partir desse conhecimento, gera regras mais complexas a cada vez, aumentando sua eficiência na resolução de problemas similares aos experimentados.

1.5 Representação do conhecimento e qualidade da aprendizagem

Podemos dizer que a representação do conhecimento tem sido uma das principais preocupações dos pesquisadores que buscam métodos para representar fatos e objetos, com a finalidade de construir programas para a resolução de problemas, que é a principal finalidade da IA. Segundo Richard [RICHARD 1990], as representações do conhecimento são construções circunstanciais realizadas dentro de um contexto particular com fins específicos sendo finalizada pela tarefa e pela natureza das decisões a serem tomadas.

Como as representações consideram o conjunto dos elementos da situação e da tarefa, elas

tornam-se muito particulares e transitórias, bastando que haja uma modificação da situação analisada ou mesmo que seja levado em conta um outro elemento da situação para que a mesma seja modificada. Sua transitoriedade acontece no cumprimento da tarefa, quando a representação é transformada em outra representação ligada a outra tarefa.

Apesar do conhecimento ser considerado construção, trata-se de uma construção permanente, sendo portanto independente da tarefa a realizar e gravado em uma memória de longo termo, permanecendo inalterado enquanto não for modificado.

Como vimos no início deste capítulo, o aspecto central da aprendizagem é *construir a representação de alguma realidade*. A representação de qualquer realidade para o mundo computacional implica invariavelmente na construção de um modelo. Um modelo computacional exige definições precisas dos seus elementos constituintes expressas de forma que o mundo computacional possa manipulá-los. Evitando as pormenorizações exigidas pela Teoria dos Modelos [DAVIS 1983], podemos avaliar as construções destas representações em termos de três dimensões [MICHALSKI 1986]:

- **Validade**: Refere-se ao grau de acuidade com a qual cada representação "absorve" a realidade; é uma medida de distância que caracteriza a precisão do mapeamento entre a representação e a realidade.
- **Efetividade**: Esse parâmetro procura capturar o aspecto de desempenho da aprendizagem, caracterizando a utilidade da representação para conseguir chegar a um determinado objetivo. Quanto mais efetiva for a representação, melhor será o desempenho do sistema.
- **Nível de abstração**: Esse critério reflete o escopo (abrangência), detalhe e a precisão dos conceitos usados nesta descrição; define o poder explicativo da representação.

A *qualidade da aprendizagem* pode ser determinada pela junção dessas três dimensões.

1.5.1 Linguagens para representação do conhecimento

O conhecimento é um elemento básico em qualquer aplicação de IA. *Conhecimento é a informação organizada e analisada de modo a tornar-se compreensível e aplicável à solução de problemas e às tomadas de decisões.* Em IA, os sistemas não são baseados em processos algorítmicos, mas em representação e manipulação simbólica. Usando símbolos, é possível criar uma base de conhecimento que contenha todos os fatos, idéias, relacionamentos e interações de um domínio limitado. A questão agora é outra: como podemos representar conhecimento na forma de símbolos e usar esse conhecimento para inteligentemente resolver problemas? A representação de tais informações pode

ser realizada através da manipulação de estruturas simbólicas[1] que representam "bits" de conhecimento. As representações podem assumir muitas formas. Cada uma destas formas é indicada para melhor ajustar-se às informações que serão manipuladas pelo sistema que as utilizará. Algumas destas descrições podem assumir formas de algoritmos, modelos de simulação, procedimentos de controle, planos, imagens, ou teorias formais gerais, dentre outras. Um problema fundamental de qualquer pesquisa em Aprendizagem de Máquina refere-se à forma e ao método utilizado para representar ou modificar o conhecimento sendo aprendido. Como sabemos que o conhecimento pode sofrer alguma mudança ou adaptação, torna-se interessante, para não dizer indispensável, identificar os componentes e as propriedades da representação que são modificáveis pelo sistema, bem como aqueles que não o são.

Em IA, o problema crucial sobre as formas (ou linguagens) de representação de conhecimento é que eles devem prover suporte para inferências. Nós não podemos representar explicitamente tudo o que o sistema poderia eventualmente precisar saber - algumas coisas devem permanecer implícitas, para que possam ser deduzidas pelo sistema como e quando necessário. Estas deduções serão suportadas por informações provenientes do conhecimento geral sobre o domínio.

Representar tudo explicitamente traria conseqüências caras, como uso de grandes quantidades de memória. Evidentemente, também seria interessante que pudéssemos realizar inferências mais e mais complexas. Porém, há uma dicotomia entre a capacidade inferencial (o que se pode deduzir) e eficiência inferencial (quão rápido se pode deduzir). Assim, podemos escolher utilizar uma linguagem onde podemos obter conclusões simples e rápidas; entretanto, inferências complexas não seriam possíveis.

1.5.2 Características das boas linguagens de representação do conhecimento

Em geral, uma boa linguagem de representação de conhecimento deve possuir pelo menos, as seguintes características:

- Deve permitir expressar o conhecimento que se deseja representar.
- Deve permitir que um novo conhecimento seja inferido a partir de um conjunto básico de fatos.
- Deve ser clara e possuir sintaxe e semântica bem definidas. Precisamos saber quais são as expressões permitidas na linguagem e o que elas significam.

[1] Esses símbolos podem ser efetivamente de naturezas diversas. Alguém poderia implementar uma máquina inteligente com fichas, por exemplo, desde que inseridas em um espaço convenientemente organizado. Felizmente, os computadores possuem uma capacidade de representação e de avaliação bem mais interessante.

1.5.3 Lógica, objetos estruturados e sistemas de produção

Em termos gerais, existem três principais abordagens para a representação de conhecimento em IA. A mais importante é a utilização de uma Lógica. Uma lógica, possui uma linguagem com sintaxe e semântica bem definidas, e considera inferências que preservam a verdade. Porém, existem alguns problemas relacionados ao seu uso para representar modelos. Por um lado, pode não ser muito eficiente - ao se tratar apenas com uma classe específica de inferências; podemos não necessitar de todo o poder de um provador de teoremas baseado em lógica, por exemplo. Por outro lado, representar algumas coisas do senso comum em uma lógica pode não ser trivial. Por exemplo, em lógica de predicados de primeira ordem, não se pode concluir que algo é verdadeiro e pouco tempo depois decidir que aquilo era falso. Isso leva a uma contradição da qual se pode provar qualquer coisa! Poderíamos decidir usar lógicas mais complexas que permitem esse tipo de raciocínio, tais como as lógicas temporais e lógicas modais. Porém, outra abordagem é abandonar as restrições impostas pelo uso de uma lógica e usar uma linguagem menos clara, mas que possua uma representação de conhecimento mais flexível [MICHALSKI 1986].

Entre as alternativas à Lógica estão os *objetos estruturados* e os *sistemas de produção*. A idéia de objetos estruturados é representar conhecimento como uma coleção de objetos e relações; as relações mais importantes são as relações de *subclasse* e relações de *instanciação*. A relação de subclasse informa que determinada classe é uma subdivisão de uma outra classe, enquanto a relação de instanciação diz que algum indivíduo pertence a alguma classe. Um *Sistema de Produção* consiste em um conjunto de regras se-então, uma memória de trabalho e uma estrutura de controle. A memória de trabalho contém os fatos que estão em determinado momento, sendo mantidos como válidos, enquanto as regras se-então tipicamente especificam se certas condições são observadas ou validadas. A estrutura de controle aplica as regras de produção à memória de trabalho. As Regras de Produção capturam conhecimento procedural de uma maneira simples, modular.

1.6 Métodos de Aprendizagem de Máquina

As pesquisas em Aprendizagem de Máquina resultam em metodologias que podem construir novos conhecimentos, ou mesmo aumentar o conhecimento existente pela introdução de novas informações. É importante notar que apesar das diferenças nas formas da entrada de informação e estrutura de conhecimento, a maioria dos métodos de aprendizagem de máquina baseia-se nos mesmos objetivos, buscando encontrar os mesmos requerimentos. Alguns desses requisitos são admitidos como gerais, descritos como segue [CARBONELL 1986]:

- **Generalidade**: Implementar aprendizagem de máquina para tratar com grande variedade

- Regras de Produção,
- Expressões Formais Baseadas em Lógica,
- Grafos e Redes,
- Frames e Estruturas de escaninhos-e-preenchimento,
- Programas de Computador,
- Representações Múltiplas,
- Taxonomias.

1.7 Resumo do capítulo

Apresentamos nesse capítulo, uma visão geral sobre os dois principais conceitos que envolverão este trabalho: Inteligência Artificial e Aprendizagem de Máquina. A compilação dos vários conceitos de IA não é uma tarefa simples; a discussão sobre a caracterização de comportamento inteligente similar ao humano em sistemas computacionais é simultaneamente empolgante e intrigante. Vimos que a IA não tem por finalidade adotar modelos de comportamento inteligente humano, mas sim prover as máquinas com capacidades de adotar atitudes mecânicas que nós utilizaríamos na solução de problemas. Alguns dos sistemas de IA citados ilustram a possibilidade de desenvolvimento de sistemas que têm o objetivo não de modelar capacidades cognitivas humanas realisticamente, mas de usar métodos computacionais para compensar limitações humanas. Os seres humanos não são naturalmente eficientes para realizar pesquisa em bancos de dados volumosos, nem para manipular conjuntos de regras com muitas variantes com o objetivo de fazer predições. Pesquisas das ciências cognitivas mostraram que os seres humanos têm uma tendência para enfocar muito rapidamente uma hipótese antes de fazer uma busca sistemática no espaço de hipóteses [GARDNER 1983]. Programas de descoberta que são mais sistemáticos e mais completos que os humanos são de grande ajuda aos cientistas. Isto tem sido alcançado e é uma conseqüência do uso da IA [DARDEN 1997].

Vimos também neste capítulo as técnicas e disciplinas utilizadas pela IA e até mesmo suas fraquezas e limitações. Apresentamos também o conceito de Agentes Inteligentes que, muitos pesquisadores encaram como a "nova face" da IA. Quanto à Aprendizagem de Máquina, vimos que esta área que nasceu contemporânea à IA, intercede as fronteiras de várias áreas do conhecimento; principalmente as da própria IA e a das ciências cognitivas. A definição de Aprendizagem de Máquina também é bastante diversificada; não conseguimos encontrar dentre as demais áreas outra mais próxima da IA; isto é notório até no senso comum, onde a capacidade de aprender é tomada

como uma medida da inteligência. Vimos como as máquinas aprendem e o tipo de conhecimento que é aprendido. No próximo capítulo, apresentaremos um tipo especial de aprendizagem de máquina, que será relevante para a consolidação desse trabalho; A *Aprendizagem Estrutural*.

Capítulo 2

Aprendizagem Estrutural

2.1 Sistemas de Aprendizagem de Máquina

O processo de classificar coisas é tão freqüente no quotidiano do ser humano que o mesmo mal chega a perceber que o realiza. A *classificação* permeia a quase totalidade das nossas atividades, principalmente as tarefas relacionadas à resolução de problemas. E não poderia deixar de ser assim. Classificar é o processo de separar indivíduos, coisas e fenômenos, tomando como base algumas de suas características comuns. Os sistemas de aprendizagem utilizam mecanismos de classificação como base para sua operação. A classificação para um sistema deste tipo é o processo de atribuir a uma informação recebida (entrada do sistema) um nome que designa a classe à qual pertence. Este processo implica em estabelecer uma descrição para estas classes. Como as classes se apresentam de formas diferentes dependendo do uso ou aplicação a que são submetidas, tais descrições poderão ser realizadas de várias formas.

Para que um sistema classificador possa atribuir a uma determinada entidade uma classe, é necessário que esta classe já tenha sido definida antes do processo de classificação ser iniciado. A definição das classes poder ser feita de várias maneiras, entre as quais [RICH 1991]:

- **definir cada classe como uma soma ponderada das características relevantes do domínio**: Isto pode ser feito isolando-se um conjunto de características relevantes para o domínio da tarefa em um conjunto C, digamos

  $$C = \{C_1, C_2, ..., C_n\}$$

  onde cada C_i representa o valor de um parâmetro relevante. Associamos a cada C_i um valor p_i (discreto ou contínuo), como um peso que exprime a existência ou a importância da característica associada. Podemos então considerar uma classe como sendo uma função do tipo:

  $$f(C) = \sum_{i=1}^{n} C_i p_i$$

- **definir classes estruturalmente**: Neste caso, também isolamos um conjunto de características relevantes dos componentes; cada classe é então definida como uma estrutura

composta por essas características.

Podemos notar que o método humano de classificação deve estar bem mais próximo da definição estrutural. Existem vantagens e desvantagens em ambos os métodos; podemos adiantar que a utilização de uma função de avaliação é geralmente mais eficiente. Não possui entretanto, a flexibilidade e extensibilidade da definição estrutural.

Independentemente da maneira como as classes estão descritas, a definição de classes está relacionada ao problema da modelagem do domínio, que não é uma tarefa trivial; Imagine estabelecer valor para as características relevantes e seus respectivos pesos para um determinado conjunto C. Se n pessoas realizassem esta tarefa independentemente, poderíamos ter n diferentes funções para uma mesma classe. Fazendo o mesmo para o segundo caso, poderíamos ter a omissão de alguns parâmetros relevantes ou ao contrário, considerar características que não seriam na verdade relevantes para a classe. Por esse motivo, a idéia de produzir um programa de classificação que consiga desenvolver sua própria definição de classe é tão atraente! Esse processo de construção de classes denomina-se *indução*.

2.2 Aprendizagem Indutiva

A aprendizagem indutiva é caracterizada pelo uso da indução nos processos de classificação. As técnicas ou métodos de indução a serem utilizadas para a construção de classes dependem da maneira como estão descritas essas classes. Algumas técnicas utilizadas para as formas de descrição já vistas são:

1. **Técnicas ou métodos de indução para a descrição por função de avaliação:**
   - Ajuste de coeficientes
2. **Técnicas ou métodos de indução para a descrição estrutural:**
   - Programa de Aprendizagem de Winston [WINSTON 1975]
   - Espaços de versões [MITCHELL 1977, 1978]
   - Árvores de Decisão (ID3) [QUINLAN 1986]
   - INNE [LIQUIÈRE 1990, 1992]

Uma distinção que identifica dois principais tipos de sistemas indutivos de aprendizagem pode ser estabelecida [MICHALSKI 1983]:

- **Aprendizagem a partir de Exemplos,** também chamada *Aquisição de Conceitos*. Neste

tipo de sistema de indução, o sistema identifica caracterizações de alguns objetos (situações, processos, etc.) pré-classificados por um instrutor em uma ou mais classes (ou conceitos). A hipótese induzida pelo sistema pode ser caracterizada como uma regra de reconhecimento conceitual, de forma que se um objeto for reconhecido por esta regra, então este objeto pertence à classe ou conceito respectivo.

- **Aprendizagem por Observação**, também conhecida por *Descrição Generativa*. Neste caso, o objetivo é determinar ou estabelecer uma descrição geral (teoria) que caracteriza uma coleção de observações. Ou seja, este tipo de sistema produz descrições que especificam propriedades de objetos pertencentes a determinada classe.

Aprender a partir de exemplos é uma área importante e ainda básica da aprendizagem de máquina. Nesta abordagem, um sistema de aprendizagem recebe um ou mais exemplos contendo os fatos e desempenho no domínio da aplicação. Baseado nestes fatos, o sistema de aprendizagem generaliza declarações que são independentes de qualquer exemplo individual.

Os sistemas de aprendizagem a partir de exemplos existentes são geralmente formados segundo as seguintes suposições [DELGRANDE 1987]:

1. o domínio é descrito como um conjunto de atributos atômicos;
2. alguns subconjuntos finitos dos atributos atômicos são dados como exemplo;
3. os exemplos são inicialmente reconhecidos como positivos ou negativos.

A primeira suposição é discutível pois não existe unanimidade na aceitação de que o mundo real esteja moldado em características básicas ou elementares. Para o nosso propósito nesta dissertação, porém, isto sugere a habilidade de perceber o mundo em termos de um vocabulário básico. Dada esta evidência, parece razoável assumir que entradas para sistemas de aprendizagem a partir de exemplos possam ser construídas a partir de um vocabulário de características básicas. Este conjunto de características denomina-se *atributos*.

A segunda e a terceira suposição não são tão discutíveis. Entretanto, nenhum dos sistemas de aprendizagem existentes examinam uma questão que é fundamental: qual é a estrutura básica de um exemplo? Invariavelmente, os exemplos são fornecidos como composições corretas das características básicas. Isso porém, nem sempre condiz com a realidade.

2.3 Aprendizagem na Presença de Ruído

O ruído pode ser definido como uma perturbação ou distúrbio, que ocorre em geral de maneira ale-

atória, reduzindo ou obscurecendo a qualidade de uma determinada informação. No contexto da aprendizagem de máquina a partir de exemplos, existem dois tipos de ruído:

- **Ruído de Classe**: Ocorre aleatoriamente alterando os valores das classes.
- **Ruído de Atributo**: Ocorre aleatoriamente alterando os valores dos atributos

Em sistemas de aprendizagem a partir de exemplos, estes erros têm sua origem fora do sistema; o conjunto de objetos da amostra já possui os erros (dados imperfeitos, imprecisos ou incoerentes) tanto nos exemplos positivos como nos exemplos negativos. Em alguns casos pode haver dados experimentais cujos atributos tenham valores com variações importantes (estatística experimental). Isto dificulta a tarefa de aprendizagem, pois no caso da presença de ruído, um exemplo tomado como positivo pode ser na verdade um exemplo negativo e vice-versa. Devemos então, neste caso, descartar a indução de conceitos sobre todos os exemplos positivos e nenhum negativo. Procedendo desta maneira, um sistema de aprendizagem a partir de exemplos portadores de ruído, realizará classificações sem considerar o erro presente em alguns dos exemplos, na esperança de que estes exemplos que contenham o erro sejam, de fato, exemplos positivos da classe ou um conceito aprendido. Desta maneira, este sistema teria se comportado como se tivesse ignorado os exemplos com erro (neste caso, o ruído), ou, numa visão mais otimista, os erros encontrados nos exemplos não teriam sido suficientes para descaracterizá-los.

2.4 SBL e EBL

Na aprendizagem a partir de exemplos, existem dois principais paradigmas, que se diferem pela maneira como os exemplos são manipulados:

- **Aprendizagem Baseada em Similaridades (SBL – "Similarity Based Learning").** A Aprendizagem Baseada em Similaridades processa dados de um conjunto de exemplos de instâncias e de classes (ou categorias) às quais pertencem as instâncias. As instâncias são usualmente descritas em vetores de características. Para um sistema de reconhecimento, as classes podem ser constituídas de coisas naturais ou de conjuntos de estados do mundo onde alguma ação possa ocorrer. O resultado obtido por tais sistemas é alguma representação operacional das classes. É possível então decidir a qual classe pertence uma nova instância a partir de um procedimento bem formado, da descrição aprendida e desta nova instância. Isto é feito através do exame das características comuns e distintas dos elementos do conjunto de treinamento. Ex: ID3 [QUINLAN 1983] e Espaço de Versões [MITCHEL 1977].
- **Aprendizagem Baseada em Explicação (EBL - "Explanation Based Learning").** Em

contraste à SBL, que utiliza vários exemplos de treinamento e aprende com base nas regularidades (ou similaridades) encontradas, a aprendizagem baseada em explicação, ou EBL, [DEJONG 1986; MITCHEL 1986] parte do princípio que se pode aprender bastante a partir de um único exemplo. Nesta abordagem, a aprendizagem é realizada a partir de um único exemplo de treinamento, de uma descrição de alto nível sobre o objetivo da aprendizagem (conceito-objeto), de uma descrição sobre os conceitos utilizáveis (Critérios de operacionalidade) e de um conjunto de regras que descrevem relacionamentos entre objetos e ações em um domínio (Teoria do Domínio). Utilizando estas informações, EBL elabora uma generalização do exemplo de treinamento. Esta generalização deve ser suficiente para descrever o conceito objeto e deve seguir sempre os critérios de operacionalidade, que funcionam como uma explicação do que deve ser feito. Isto é sempre usado no contexto de sistemas solucionadores de problemas onde cada sessão de resolução de problema gera uma explicação que justifica a resposta para a questão.

2.4.1 Aprendizagem Baseada em Similaridade × Aprendizagem Baseada em Explicação

Vamos agora enumerar alguns pontos que diferenciam as abordagens SBL e EBL.

Normalmente um programa de aprendizagem baseado em SBL requer um conjunto de treinamento substancialmente grande (Embora existam abordagens de SBL utilizando conjuntos pequenos [WEISS 1978]). Quanto maior este conjunto, mais refinada será a aprendizagem [RICH 1994]. Considera-se que a generalização de similaridades leva a conceitos que podem ser aplicados em um novo problema. Usualmente, o processo de aprendizagem não utiliza qualquer conhecimento relacionado com o domínio para supervisionar ou guiar sua operação.

Ao contrário dos SBL, os sistemas baseados em explicação utilizam um único exemplo, constróem uma explicação sobre como os vários componentes se relacionam entre si e então generalizam as propriedades de vários componentes do exemplo enquanto a explicação permanecer válida. Podemos considerar que um sistema deste tipo exige um grande conhecimento do domínio da aplicação, o que geralmente minimiza a necessidade da aprendizagem.

Estas duas abordagens não são necessariamente exclusivistas. Existem sistemas que combinam EBL com SBL. O sistema de aprendizagem OCCAM [MEEHAN 1987] por exemplo, generaliza uma descrição conceitual a partir de vários exemplos (característica SBL), utilizando um mecanismo de explicação para validá-la (característica EBL).

2.4.2 Aprendizagem a partir de exemplos

A aprendizagem a partir de exemplos é um caso especial da aprendizagem por indução. Neste tipo de aprendizagem, é dado um conjunto de exemplos e contra-exemplos de um conceito. O aprendiz induz a descrição de um conceito geral que descreve todos os exemplos positivos e nenhum dos exemplos negativos. A quantidade de inferência realizada pelo aprendiz é bem maior que a da aprendizagem por instrução, na qual nenhum conceito geral é fornecido pelo instrutor, e é maior ainda que na aprendizagem por analogia, onde nenhum conceito similar é fornecido como "raíz".

A aprendizagem a partir de exemplos pode ser caracterizada de acordo com a fonte dos exemplos:

- **A fonte é um instrutor**. Neste caso, a fonte já conhece o conceito e gera sequências de exemplos deste conceito para o aprendiz; são escolhidos inicialmente, exemplos considerados os mais úteis possíveis, ou os mais relevantes. Se o instrutor conhece ainda o estado atual do aprendiz, os exemplos podem ser selecionados de maneira que a convergência para os conceitos a serem ensinados seja melhorada.
- **A fonte é o próprio aprendiz.** Neste caso, o aprendiz tem consciência do seu estado de conhecimento, mas não conhece o conceito a ser adquirido. O aprendiz gera então instâncias deste conceito, utilizando uma base de informações que ele acredite serem necessárias para discriminar entre as possíveis descrições do conceito. Essas instâncias são então classificadas como instâncias positivas ou negativas do conceito; para isso, o aprendiz deve recorrer ao auxílio de uma entidade externa, pois como ele não conhece o conceito, deve interrogar alguém que o conheça.
- **A fonte é o ambiente externo**. Neste caso, o processo de geração dos exemplos é operacionalmente aleatório, pois o aprendiz deve basear-se em observações relativamente não controladas.

Podemos ainda caracterizar esta aprendizagem considerando o tipo de exemplos apresentados ao aprendiz:

- **Apenas exemplos positivos estão disponíveis**. Os exemplos positivos fornecem as instâncias do conceito a ser adquirido. Eles não fornecem informações para a prevenção do problema da generalização em larga escala do conceito inferido. Neste tipo de situação, este problema pode ser evitado usando a mínima generalização necessária, ou então utilizar um conhecimento previamente adquirido sobre o domínio (restrição da inferência).
- **Exemplos positivos e negativos estão disponíveis.** Nesta situação, os exemplo positivos

forçam a generalização e os negativos possibilitam evitar a generalização em larga escala.

2.4.3 Aprendizagem Incremental e Aprendizagem Não-Incremental

A aprendizagem a partir de exemplos pode utilizar basicamente dois tipos de conjuntos de exemplos de treinamento (domínio):

- **Estático**, no qual o método de aprendizagem utiliza apenas um único conjunto de treinamento, que é fixo, invariável, e
- **Dinâmico**, no qual o método utiliza diferentes conjuntos de treinamento, que apresentam alterações a cada vez que são utilizados.

Podemos intuitivamente deduzir que a aprendizagem realizada sobre domínios dinâmicos é bem mais abrangente e complexa. Vamos agora considerar o seguinte problema. Suponhamos que tenhamos gerado uma determinada base de conhecimento a partir da aprendizagem realizada sobre um determinado conjunto de treinamento, e agora desejamos apresentar novos exemplos para enriquecer a nossa base de conhecimento existente; Temos duas possibilidades, definidas pelo algoritmo de apresentação de exemplos:

- **Não-Incremental**. Neste caso, a base de conhecimento existente é descartada à medida em que novos conjuntos de treinamentos são apresentados. Os conjuntos de treinamento são formados pelos exemplos já processados e também pelos novos exemplos, como mostra a fig. 2.1. Aparentemente, este algoritmo não é muito eficiente, pois terá que reaprender muitas das coisas que já havia aprendido. Veja por exemplo, o caso de apresentarmos um novo conjunto de exemplos de tamanho muito reduzido: o algoritmo construiria uma nova base de conhecimento quase idêntica à anterior. Alguns autores chegam a afirmar que o uso deste algoritmo torna-se inviável quando se utiliza um grande e diversificado número de exemplos [MICHALSKI 1985].
- **Incremental**. Nesta situação, o conjunto de treinamento é apresentado juntamente com a base de conhecimento já existente. Desta maneira, existe o incremento da base de conhecimento a partir das novas observações notadas no novo conjunto de treinamento apresentado como mostra a fig. 2.2. Um sistema que utilize este algoritmo deve formar uma ou mais hipóteses do conceito (ou conceitos) consistentes com os dados disponíveis, e subsequentemente, refinar a hipótese após considerar exemplos adicionais. Esta abordagem é bastante interessante, pois permite a utilização de conceitos parcialmente aprendidos e permite ao instrutor focalizar sobre os aspectos básicos do novo conceito antes dos aspectos menos importantes.

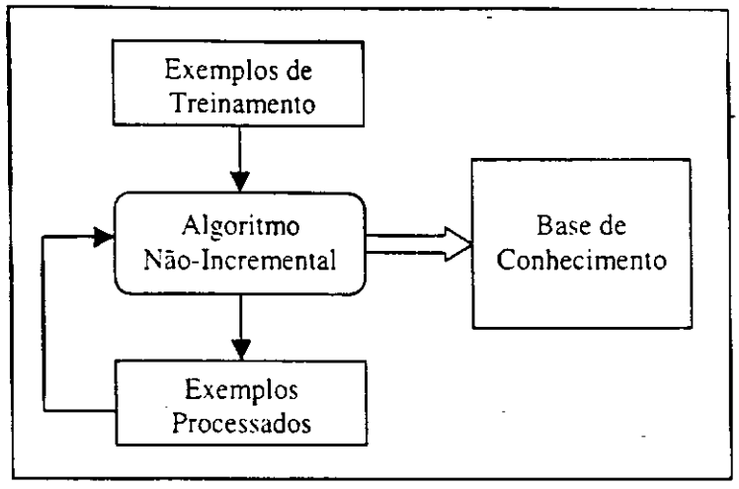


Fig. 2.1 - Algoritmo Não-Incremental

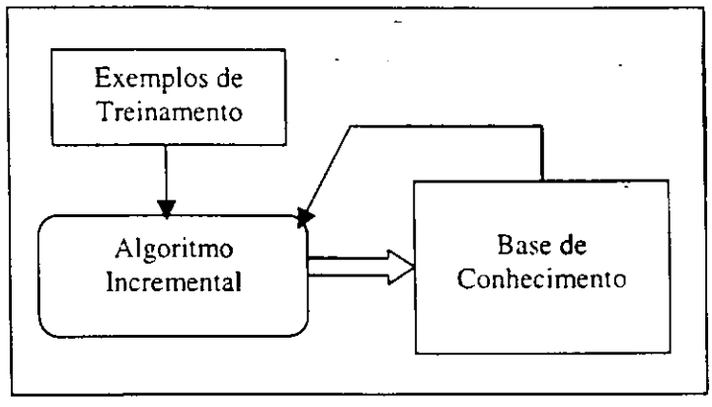


Fig. 2.2 - Algoritmo Incremental

Curiosamente, considerando o modo humano de aprender, análises de desempenho mostram que os algoritmos não-incrementais são, na maioria das vezes, mais eficientes que os algoritmos incrementais; para um único processamento utilizando domínios estáticos ou dinâmicos, os algoritmos incrementais são mais eficientes que os não-incrementais; a partir do segundo processamento, os algoritmos incrementais são mais eficientes sob certas situações utilizando domínios dinâmicos, mas de maneira geral os não-incrementais são mais eficientes [BEZERRA 1995].

2.5 Aspectos importantes da aprendizagem a partir de exemplos

O processo de aprendizagem indutiva pode ser visto como uma busca por descrições gerais plausíveis (afirmações indutivas) que generalizem a informação inicial fornecida e sejam úteis para predizerem novas informações. Para que um programa de computador formule tais descrições, uma linguagem de descrição apropriada deve ser utilizada. Para qualquer conjunto de informação de entrada e para qualquer linguagem não trivial de descrição, pode ser formulado um número considerável de afirmações indutivas. Estas afirmações formam um conjunto de descrições parcialmente

ordenadas de generalidade relativa [MITCHEL 1977]. Os elementos mínimos desse conjunto são as descrições mais específicas das informações de entrada na linguagem dada, e os elementos máximos são as descrições mais gerais destas informações. Os elementos desse conjunto podem ser generalizados a partir das descrições mais específicas e repetidamente, aplicando regras de generalização para descrições mais gerais.

Para ver a indução como uma busca através de um espaço de descrições generalizadas, os seguintes aspectos da aprendizagem devem ser considerados:

- **Representação**. Que linguagem de descrição é empregada para expressar os exemplos inicialmente apresentados e formular afirmações indutivas? Quais são as possíveis formas de afirmações que um método pode aprender? Que operadores são utilizados para estas formas?
- **Tipo de descrição procurada**. Para qual propósito estão sendo formuladas as afirmações indutivas? Que suposições o método de indução faz sobre o processo subjacente que gera a informação?
- **Regras de generalização**. Que tipo de transformações são realizadas sobre as informações inicialmente apresentadas e as descrições intermediárias?
- **Indução Construtiva**. O processo de indução muda o espaço de descrição? Ou seja, produz novos descritores que não estavam presentes nos eventos inicialmente apresentados?
- **Estratégia de Controle**. Qual é a estratégia usada para busca no espaço de descrição? Orientada a dados, orientada a modelos ou mista?
- **Abordagem geral ou específica**. O método é orientado para resolver uma classe geral de problemas ou para resolver problemas de um domínio específico?

2.6 Exemplos de programas de aprendizagem estrutural a partir de exemplos

Apresenta-se a seguir, uma breve descrição de alguns dos principais programas de aprendizagem a partir de exemplos que utilizam descrição estrutural.

2.6.1 O Programa de Aprendizagem Estrutural de Conceitos de Winston

Patrick Winston é considerado um dos pioneiros em aprendizagem de máquina a partir de exemplos. Ele desenvolveu um programa clássico de aprendizagem estrutural de conceitos com as ca-

racterísticas mostradas na Tab. 2.1 [WINSTON 1975]:

Domínio	*Mundo dos blocos*
Meta	*Construção de representações das definições de conceitos no mundo dos blocos*
Estrutura do conhecimento	*Redes semânticas* [QUILLIAN 1966]
Ponto principal	*Aprendizagem de conceitos, analisando as diferenças que aparece numa seqüência de observações sobre a estrutura de objetos que são instâncias ou aproximações do conceito a ser aprendido.*
Técnica	*Procedimentos que aprendem descrições de classe de exemplos positivos e exemplos negativos*
Heurísticas	*Convenções implícitas entre o instrutor e o aprendiz*

Tab. 2.1 - Características do Programa de Aprendizagem de Winston

O programa de Winston aprende conceitos estruturados, como por exemplo, os conceitos clássicos de "casa", "tenda" e "arco". Nessa abordagem, podem ser considerados como portadores de ruído, exemplos definidos em termos de "erros por pouco" e de "aproximações" (Fig. 2.3) para cada conceito. Aproximação é um objeto que, apesar de não ser uma instância do conceito em questão, apresenta uma forte semelhança com o mesmo.

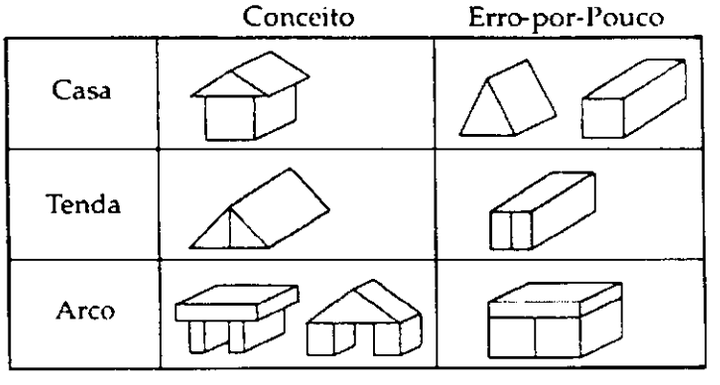


Fig 2.3 - Conceitos do mundo dos blocos

A descrição em redes semânticas é bastante simples. Cada nodo da rede representa um conceito componente da estrutura; os arcos ou setas indicam a relação que existe entre os nodos. Na fig. 2.4 é apresentada uma rede semântica representando o conceito de "casa". A relação "É_um" é uma relação especial; ela indica a classe à qual pertence o componente da estrutura.

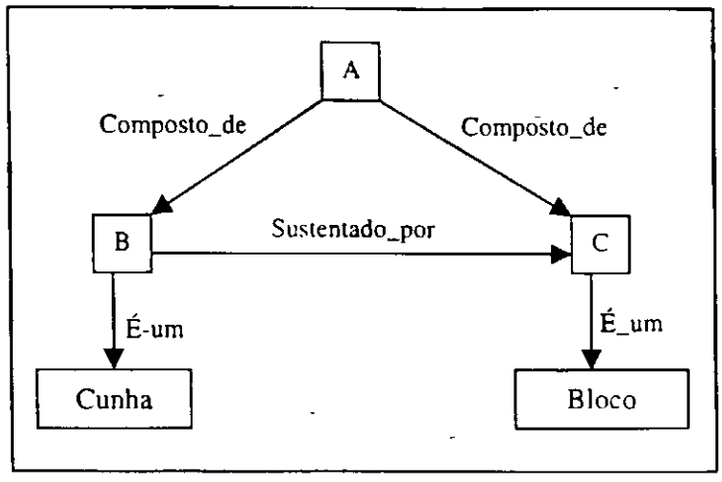


Fig. 2.4 - Rede semântica para a descrição estrutural de "casa"

2.6.1.1 Abordagem básica do programa de Winston

O programa inicia com uma descrição estrutural de uma instância conhecida do conceito, chamada *descrição inicial*. A definição do conceito já é ligada a esta primeira descrição. Durante a aprendizagem, a descrição inicial é aumentada através da apresentação de novas instâncias, que fornecem cada uma informações que indicam quais vínculos são importantes. A descrição aumentada é chamada *modelo evoluído*. As informações fornecidas pelas outras instâncias conhecidas do conceito modificam a descrição inicial a partir de duas operações:

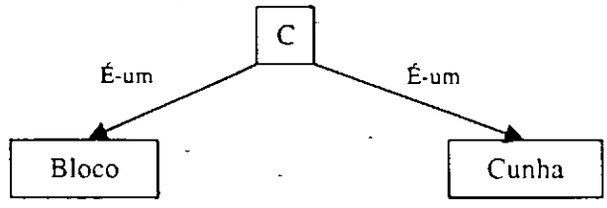
- **Generalização por inclusão.** Esta operação ocorre quando é apresentada uma instância cuja descrição é um exemplo autêntico do conceito, mas um de seus componentes não consta no modelo evoluído. Nesse caso, o modelo é aumentado para que possa ser generalizado também para essa nova instância. Um modelo é denominado pobre quando apresenta um baixo nível de generalização, a exemplo de quase todas as descrições iniciais. Um modelo é dito ser rico quando alcança um nível aceitável de generalização. A caracterização da aprendizagem se dá pelo gradual enriquecimento do modelo.

- **Generalização por restrição**: Esta operação se dá quando no lugar de uma instância do conceito apresenta-se ao sistema uma *aproximação* ou *erro por pouco*. Uma vez verificada a diferença, ao invés de expandir o modelo para suportar a nova instância, restringe-se a definição para que a diferença detectada seja especificamente excluída do modelo.

2.6.1.2 Exemplo do programa de Winston

Observemos o seguinte exemplo. Vamos supor que apresentamos ao programa a descrição estrutural do conceito de um arco, conforme descrito na fig 2.5a. Inicialmente, essa descrição é tomada

como a descrição do conceito; trata-se de um modelo pobre, por tratar-se evidentemente de um modelo resultante de uma única entrada. Vamos agora supor que uma outra instância seja apresentada, e que sua descrição corresponda à da fig. 2.5b; observe que a única diferença reside na especificação do componente "C"; no primeiro caso, tratava-se de um bloco, e agora, trata-se de uma cunha. Ambas as descrições são, com efeito, exemplos do conceito arco. O que o programa faz é aumentar o modelo para admitir a nova descrição. Como o programa aumenta o modelo? Ele pode simplesmente bifurcar o nodo "C", criando um novo arco para "Cunha", associando uma função "OU", da seguinte maneira:



O programa pode opcionalmente, apresentar uma solução mais elegante, acompanhado a hierarquia dos elementos "bloco" e "cunha", e em encontrando, assumir este componente, como mostra a fig. 2.5c, que encontrou a hierarquia "objeto" que inclui ambos. Esta operação é um exemplo da generalização por inclusão. Neste ponto, o modelo já está um pouco mais rico, podendo reconhecer exemplos mais complexos que o primeiro. Observe que se a solução for aplicada, o programa reconhecerá como um arco qualquer estrutura que possua dois blocos e tenha alguma coisa que pertença à classe "objeto" sobre estes blocos.

Agora suponhamos que a próxima entrada seja uma aproximação, como a mostrada na fig. 2.3. O programa identifica a diferença entre o modelo evoluído atual e a nova entrada, retornando a relação entre os componentes "B" e "D": Os mesmos possuem a relação "toca", que difere do modelo evoluído, onde esta relação é "não-toca", como a entrada representa uma aproximação ou erro por pouco, o modelo deve ser modificado para não acatar este tipo de relação entre os componentes "B" e "D", identificada como a restrição do modelo. Isto é feito, enfatizando a relação já existente "não-toca", modificando a mesma para "necessariamente-não-toca".

2.6.1.3 Problemas relacionados com a abordagem de Winston

A abordagem de Winston apresenta alguns inconvenientes, a saber:

- O instrutor tem que guiar o programa de aprendizagem, certificando-se de que as representações sejam suficientes e provendo não só exemplos positivos, mas também exemplos negativos e aproximações.

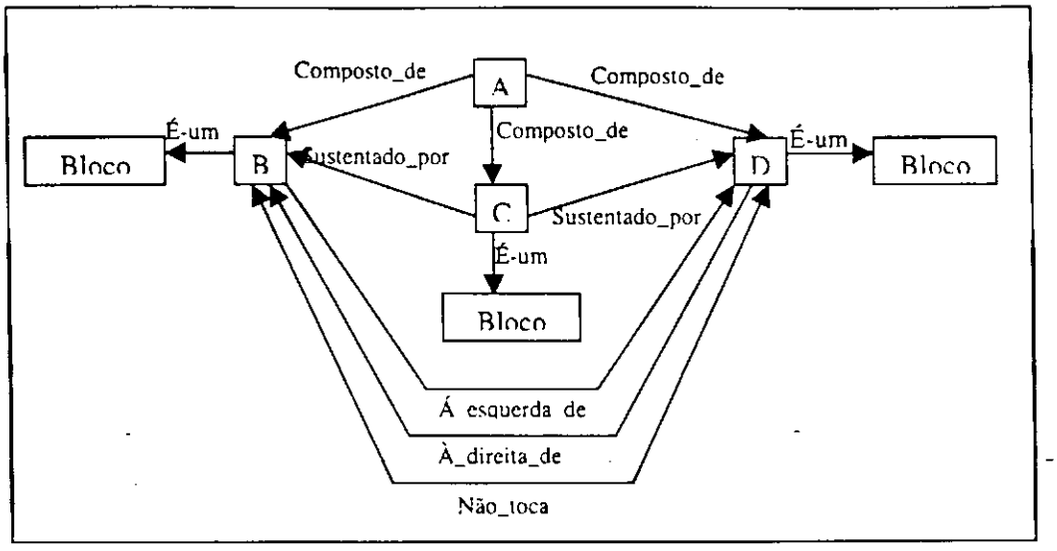


Fig. 2.5.a - Rede semântica para a descrição estrutural de "arco"

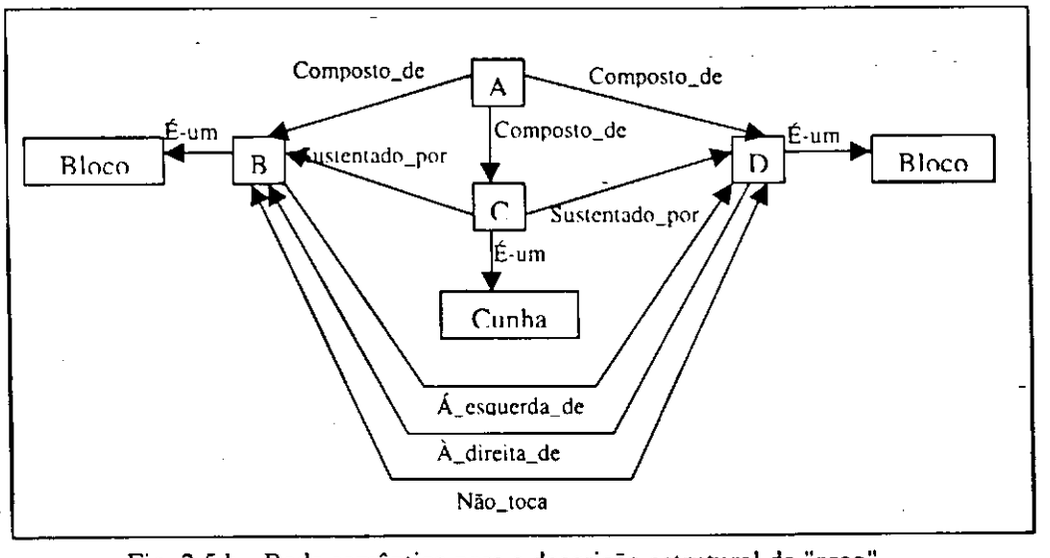


Fig. 2.5.b - Rede semântica para a descrição estrutural de "arco"

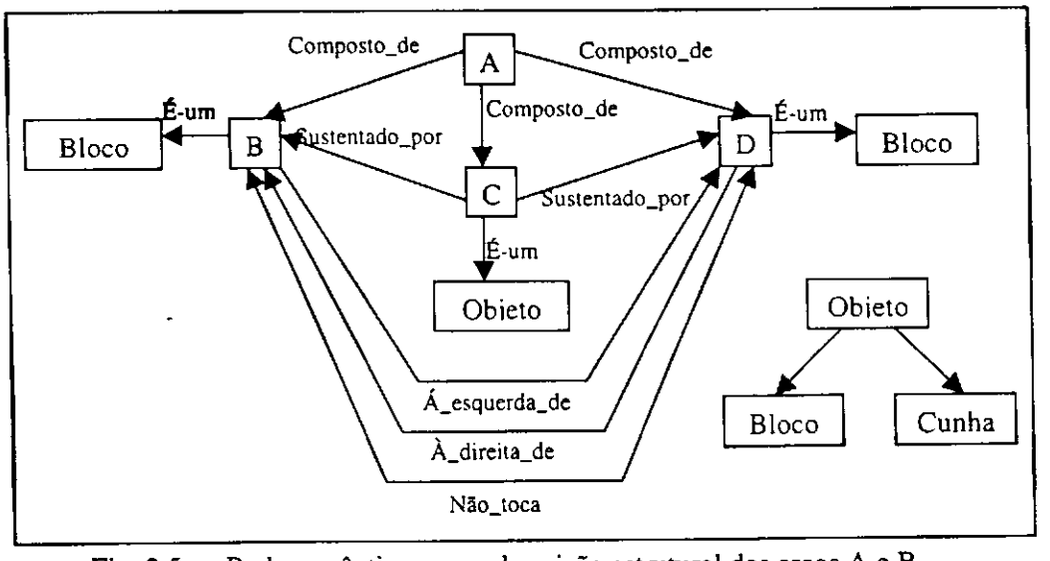


Fig. 2.5.c - Rede semântica para a descrição estrutural dos arcos A e B

- A seqüência de exemplos deve ser cuidadosamente escolhida. O ideal é construir a descrição inicial do modelo a partir de exemplos positivos e de maneira que os exemplos apresentados posteriormente sejam estruturalmente mais complexos que os antecedentes. Se o modelo inicial for construído a partir de uma aproximação, e os exemplos subsequentes não forem bem selecionados, o modelo resultante após estes primeiros exemplos poderá ser desinteressante, baseando-se mais nas exceções que nas similaridades (classificação por exceção).

2.6.2 Espaço de Versões

Um outro programa clássico de aprendizagem de conceitos a partir de exemplos com representação estrutural foi criado por Tom Mitchel [MITCHEL 1977]. O objetivo é idêntico ao programa de Winston: produzir uma descrição generalizada do conceito que seja consistente com todos os exemplos positivos e nenhum exemplo negativo. A diferença principal é que o programa de Winston desenvolvia o *modelo evoluído*, que é, como vimos, uma generalização que consistia num único modelo do conceito. O programa de Mitchel não gera um único modelo; ele constrói um conjunto de descrições possíveis, desenvolvendo esse conjunto à medida em que novos exemplos e contra-exemplos do conceito vão sendo apresentados. Suas principais características estão resumidas na Tab. 2.2.

O "Espaço de versões" admite diferentes linguagens de representação, sendo que linguagens baseadas em frames são as que melhor se prestam para este fim. Os frames são ferramentas simplificadas de representação de conhecimento; um frame que descreve um automóvel poderia ter o seguinte aspecto [RICH 1991]:

Carro [#01]

Origem:	Brasil
Fabricante:	Gurgel
Cor:	Vermelho
Ano/Modelo:	1990
Tipo:	Econômico

Os frames ou "escaninhos" podem ser vistos como os locais apontados ou rotulados por cada item componente de um carro nesta estrutura, onde se possa atribuir um valor. Por isso o nome escaninho.

Meta	*Construir representações das definições de conceitos*
Técnica	*Manter um conjunto de possíveis descrições e evoluir aquele conjunto ao passo que novos exemplos e aproximações são apresentados. Alguns previnem modificações enganosas seguindo cada interpretação porquanto permaneçam viáveis.*
Restrições	*Há um número fixo de atributos e um modelo de caracterização de classe pode ser expresso como uma combinação de valores para aqueles atributos.*
Ponto principal	*A aprendizagem pela administração de modelos múltiplos. Conceitos podem ser determinado instância a instância, sem a necessidade de examinar instâncias de treinamento passadas ou descrições de conceito previamente rejeitadas.*
Heurística	*Restrições são exatas e garantem fazer o algoritmo convergir para uma solução, assim a busca heurística não é empregada.*

Tab. 2.2 - Características do Espaço de Versões

Suponha agora que cada frame possa admitir apenas os seguintes valores:

Origem:	{Brasil, Itália, E.U.A., Inglaterra, Alemanha, Japão}
Fabricante:	{Gurgel, Ford, Ferrari, Lotus, BMW, Mercedes, Toyota}
Cor:	{Vermelho, Azul, Verde, Branco}
Ano/Modelo:	{1990, 1991, 1992, 1993, 1994, 1995}
Tipo:	{Econômico, Luxo, Esportivo, Utilitário}

A escolha dos frames e de seus valores é chamada *tendência*, porque "forçam" a aprendizagem sobre um determinado conceito.

No programa de Mitchel, os conceitos podem ser definidos em termos de frames e valores associados; por exemplo: o conceito de "Carro Econômico Japonês" pode ser representado da seguinte maneira:

Carro [Econômico Japonês]

Origem:	Japão
Fabricante:	X1
Cor:	X2
Ano/Modelo:	X3
Tipo:	Econômico

Podemos notar que essa representação permite construir uma ordem parcial dos conceitos, como mostra a Fig. 2.6: Esta ordenação, seja parcial ou completa, recebe o nome de *espaço de conceitos*. Note que no topo deste espaço de conceitos estão apenas variáveis, que chamaremos *descrição nula ou hipótese nula*, e representará qualquer possível instância. Na sua base, encontram-se elementos com a descrição completa, que não contém nenhuma variável e representam todas as

possíveis instâncias de treinamento. Esta representação pode ser ilustrada como mostrado na Fig. 2.6.

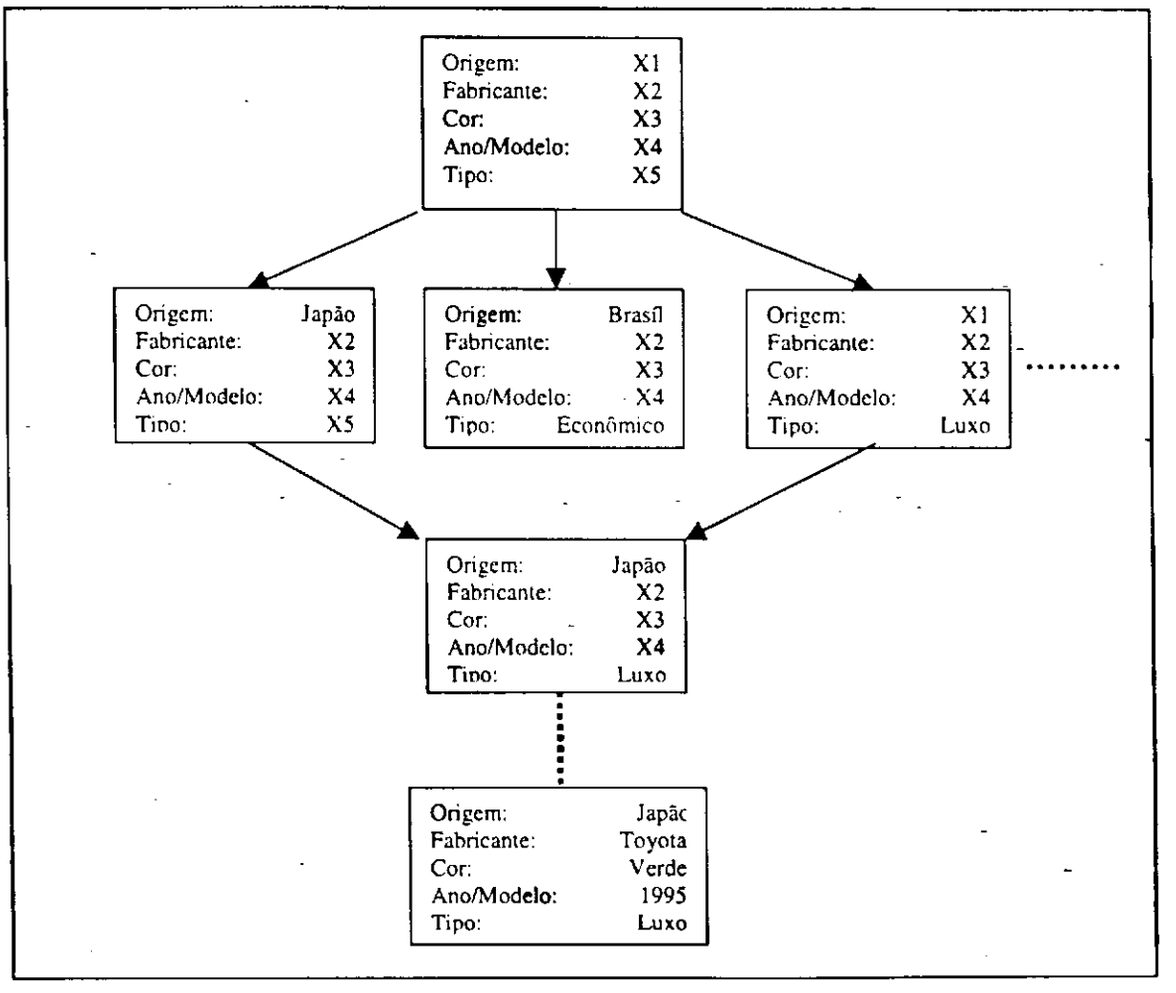


Fig. 2.6 - Ordem parcial associada aos conceitos

O triângulo apresentado na Figura 2.7 delimita uma região chamada *espaço de conceitos*, onde se encontram todas as possíveis descrições do *conceito-objeto*, ou seja, o conceito que se procura aprender. No topo do triângulo, fica a hipótese nula; na sua base, os exemplos de treinamento vistos até então. O triângulo é cortado por duas linhas que delimitam o espaço segundo a ordem de abrangência da descrição conceitual; quanto mais abrangente ou genérica, mais próximo do topo do triângulo (região G); quanto mais específica, mais próxima da base do triângulo (região E). Entre essas duas regiões, encontra-se o *espaço de versões*, onde se localiza o maior conjunto de descrições consistentes com todas as instâncias de treinamento já vistas. Dada a estrutura da fig. 2.5(a), podemos ver que mesmo antes da apresentação de qualquer exemplo, o conceito-objeto está em algum lugar do espaço de conceitos. Em um caso extremo, quando a descrição que representa o conceito-objeto admite todas as instâncias, então a definição do conceito é a hipótese nula, pois é a definição que vale para todas as instâncias; qualquer outra teria uma ou mais variáveis (escaninhos)

preenchidas, o que implicaria numa redução ou filtragem do espaço. Geralmente, o conceito-objeto procurado encontra-se em algum lugar entre os dois extremos (região V).

O princípio de funcionamento é simples. Considere a instância de treinamento positiva I+ e a instância de treinamento negativa I-. Temos que I+ $\uparrow \Rightarrow$ E $\uparrow$ e I - $\uparrow \Rightarrow$ G $\uparrow$. Isto é, sempre que se apresentem exemplos positivos, o conjunto E deve tornar-se mais genérico para absorver cada novo exemplo; cada vez que se apresentem exemplos negativos, o conjunto G torna-se mais específico. À medida em que novos exemplos positivos e negativos vão sendo apresentados, E e G tendem a se aproximar, convergindo para a redução do espaço de versões, o que significa a redução da faixa de hipóteses, que poderá ser reduzida para conter uma única descrição do conceito. O algoritmo para efetuar esta redução do espaço de versões é chamado *eliminação de candidatos* e funciona como mostrado a seguir.

2.6.2.1 Abordagem básica do programa de Mitchel

Considere o seguinte problema: "Gerar uma descrição do conceito que seja consistente com todos os exemplos positivos e nenhum exemplo negativo". O algoritmo procede da seguinte maneira:

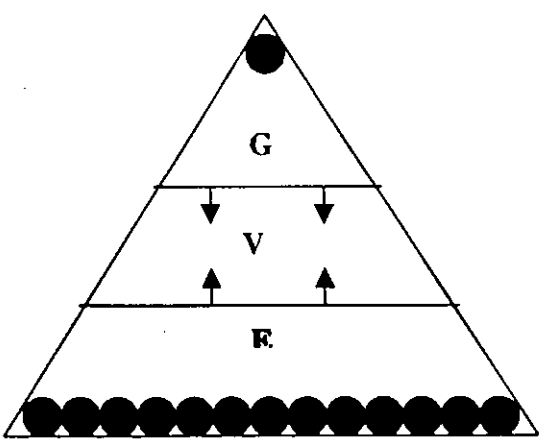


Fig. 2.7 - Espaço dos Conceitos e das Versões

1. Inicializa-se o conjunto G para conter todas as variáveis características (hipótese nula);
2. Inicializa-se o conjunto E para conter o primeiro exemplo positivo;
3. Apresenta-se um novo exemplo;

   Se for Positivo:

- Remove de G qualquer descrição que não aceite o exemplo;
- Atualiza E para conter o conjunto mais específico de descrições que aceite o exem-

plo e o próprio conjunto E.

Se for Negativo:

- Remove de E qualquer descrição que aceite o exemplo apresentado;

- Atualiza G que passa a conter o conjunto mais genérico de descrições.

4. Compare os conjuntos E e G;

Se E = G e ambos contêm apenas uma descrição:

- Pare e Retorne com a descrição – ela representa o conceito procurado!

Se E ≠ G e ambos contêm apenas uma descrição:

- Pare - os elementos apresentados no treinamento eram inconsistentes e não foi gerada nenhuma descrição

Se E e G contêm mais que uma descrição, ainda não houve a convergência - Desvia-se para o passo # 3.

2.6.2.2 Resumo das propriedades

A abordagem de Mitchel apresenta as seguintes características:

- todas as descrições de conceito que são consistentes com todos os exemplos de treinamento são encontradas;
- o espaço de versões resume as interpretações alternativas dos dados observados;
- os resultados são independentes da ordem na qual as instâncias de treinamento são apresentadas;
- cada instância de treinamento só é examinada uma única vez;
- não é necessário reconsiderar hipóteses descartadas. A representação de conjuntos limites para o espaço de versões é tanto compacta como fácil de atualizar;
- é compacto pois não se armazena toda descrição de conceito explicitamente no espaço;
- é de fácil atualização, visto que definir um novo espaço corresponde a mover um ou ambos os limites

2.6.2.3 Problemas relacionados

A abordagem de Mitchel apresenta alguns inconvenientes, dentre eles:

- o algoritmo demanda grandes exigências espaciais (grande quantidade de memória), pois implementa uma busca exaustiva em amplitude;
- dados incompatíveis (ruído), podem fazer o algoritmo "podar" prematuramente, o conceito alvo do espaço de versão, fazendo com que o conjunto G atinja um nível tão alto de especialização que o espaço de versões fique vazio;
- a aprendizagem dos conceitos disjuntivos dá margem a generalizações desinteressantes. Por exemplo, se quisermos aprender o conceito "carro europeu", a generalização resultante sobre os fatos "Alemanha", "Itália" e "Inglaterra" seria "qualquer origem". Com isso, um carro japonês seria indevidamente reconhecido pelo algoritmo.

2.6.3 O ID3

O programa "ID3" [QUINLAN 1986] utiliza uma representação de conceitos em estruturas denominadas *árvores de decisão*. Numa árvore de decisão, um objeto pertencente ao conceito pode ser identificado com o percurso a partir da raíz da árvore, até que seja alcançada uma folha. O percurso é feito respondendo às perguntas em cada nodo, cuja resposta é um caminho derivado desse nodo.

O algoritmo ID3 aprende, construindo uma árvore de decisão a partir de atributos e valores dos objetos. Uma árvore de decisão que identifica uma fruta é mostrada na fig. 2.8. Vamos analisar como esta árvore de decisão foi gerada, acompanhando o algoritmo para o exemplo dado.

Algoritmo para a construção da árvore (ID3):

1. Descrever todos os objetos em termos dos seus atributos;
2. Repetir até que todos os objetos estejam divididos em atributos simples ...

    Tomar um atributo;

    Dividir os objetos, segundo a presença ou ausência deste atributo.

De acordo com o nosso exemplo, temos os seguintes objetos e seus atributos:

Laranja:	redonda, alaranjada, macia
Maçã:	redonda, verde, macia
Pêssego:	redonda, pelugem
Maracujá:	redonda, amarela
Banana:	amarela, comprida
Pêra:	verde, ovalada

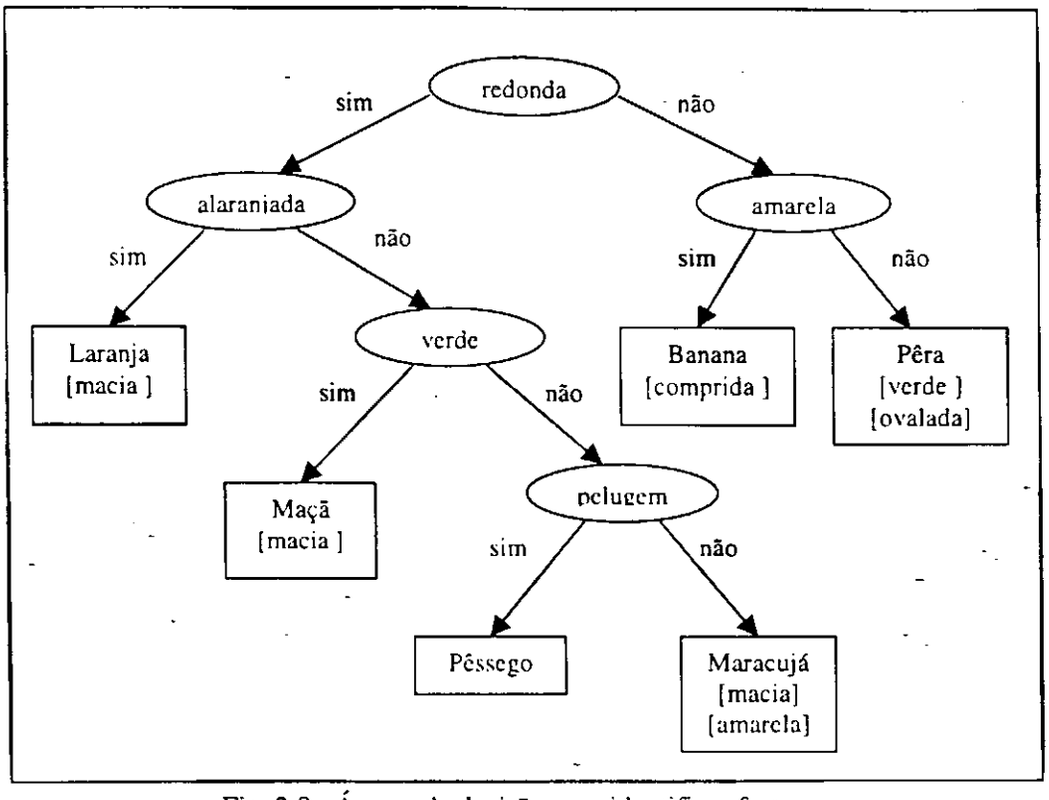


Fig. 2.8 - Árvore de decisão para identificar frutas

Toma-se o atributo "redonda" e seleciona-se os objetos que possuem e os que não possuem este atributo. Derivamos duas listas: Objetos que possuem aquele atributo e Objetos que não possuem aquele atributo.

Repetimos o procedimento para todos os demais atributos. Deve ser observado que existem outras possibilidades de combinação (árvores diferentes), em função da ordem em que os atributos são considerados.

A partir da árvore de decisão finalizada, é possível construir regras SE-ENTÃO para cada objeto pertencente à árvore, percorrendo a árvore desde a raiz até a folha contendo o objeto. A lista de regras associadas à árvore do exemplo apresentado é:

SE REDONDA [SIM] E ALARANJADA [SIM] ENTÃO LARANJA
SE REDONDA [SIM] E ALARANJADA [NÃO] E VERDE [SIM] ENTÃO MAÇÃ
SE REDONDA [SIM] E ALARANJADA [NÃO] E VERDE [NÃO] E PELUGEM [SIM] ENTÃO PÊSSEGO
SE REDONDA [SIM] E ALARANJADA [NÃO] E VERDE [NÃO] E PELUGEM [NÃO] ENTÃO MARACUJÁ
SE REDONDA [NÃO] E AMARELA [SIM] ENTÃO LARANJA
SE REDONDA [NÃO] E AMARELA [NÃO] ENTÃO PÊRA

2.6.3.1 Abordagem básica do ID3

O ID3 constrói árvores de decisão, preferindo árvores simples às complexas, a partir de um conjunto de exemplos de treinamento, escolhidos de maneira aleatória; esse subconjunto da amostra de treinamento é chamado janela. O ID3 constrói para a janela uma árvore de decisão que classifica corretamente todos os seus exemplos. A partir desse ponto, o ID3 testa a árvore gerada, tentando classificar os elementos da amostra que estavam fora da janela. Caso consiga classificar a todos, o algoritmo pára (coincidentemente, escolheu um conjunto inicial bastante representativo); caso contrário, ele acrescenta os exemplos de treinamentos não classificados aos já contidos na janela, reconstrói a árvore de decisão e continua o processo, até que a janela classifique todos os exemplos da amostra.

2.6.3.2 Vantagens e desvantagens do ID3

O algoritmo proposto por Quinlan apresenta vantagens e desvantagens que podem ser encontradas com maiores detalhes em Mongiovi [MONGIOVI 1993] e Gaines [GAINES 1993]. Dentre as principais vantagens do algoritmo podemos enumerar as seguintes:

- Mais eficiente que o espaço de versões, para espaços conceituais grandes;
- É muito rápido.

Dentre as desvantagens, citamos as mais importantes:

- A representação não é boa para análise humana, principalmente em árvores grandes;
- Não é indicada para objetos portadores de ruído;
- O conhecimento fica pulverizado por toda a árvore, ficando as informações relevantes para uma classificação subordinadas a condições por vezes irrelevantes;
- A ausência de valores nos atributos dos exemplos traz problemas importantes ao algoritmo;
- Pouca resistência ao ruído.

2.6.4 O INNE

M. Liquière [LIQUIÈRE 1990, 1992] concebeu o método de aprendizagem INNE ("Induction in Networks"), capaz de, a partir de um conjunto formado por objetos que são exemplos e contra-exemplos de um conceito, gerar como resultado dois conjuntos: um de *assimilação*, gerado a partir

de toda a amostra e que representa uma interpretação geral de todos os objetos presentes na mesma, e um de *discriminação*, que representa uma interpretação que distingue entre exemplos e contra-exemplos. A tabela 2.3 resume suas principais características.

Meta	*Construir a representação genérica de um conceito a partir de exemplos e contra-exemplos de instâncias deste conceito*
Técnica	*Busca das similaridades em conjuntos de grafos conceituais chamados Exemplos e Contra-Exemplos, verificando critérios restritivos de validade majoritária*
Restrições	*Há um número fixo de atributos e um modelo de caracterização de classe pode ser expresso como uma combinação de valores para aqueles atributos.*
Ponto principal	*A aprendizagem admite objetos portadores de ruídos; O resultado da aprendizagem pode ser interpretado em termos gerais sobre a amostra, ou admitindo apenas os exemplos sem ruído*
Heurística	*Restrições são definidas pelo usuário que deve possuir um bom conhecimento do domínio*

Tab. 2.3 - Características do INNE

O problema pode ser visto como segue: Sendo um conjunto E de grafos conceituais ditos exemplos e C um outro conjunto de grafos ditos contra-exemplos, encontrar um conjunto de classes de equivalência tal que os grafos dessas classes sejam fatos majoritariamente válidos sobre a amostra <E,C>.

Tomemos o exemplo clássico da aprendizagem do conceito *arco* a partir de uma amostra, considerando os objetos mostrados na fig. 2.9, onde podemos identificar três exemplos e um contra-exemplo (portador de ruído) deste conceito. A representação desses objetos como grafos conceituais é mostrada na fig. 2.10

2.6.4.1 Abordagem básica do INNE

O método consiste nos seguintes passos. Inicialmente, constrói-se uma classe de equivalência composta de caminhos de comprimento 0 (relações e conceitos) presentes na amostra. Para exemplificar este primeiro passo, vamos considerar a amostra apresentada na Fig. 2.10. Um critério de validade é inicialmente estabelecido; podemos estabelecer neste exemplo, como critério de validade, considerar apenas instâncias de classe que ocorram em mais de 2 exemplos da amostra, teremos o seguinte resultado:

E_0 = {<é-um>, <forma-estável>, <objeto>, <status>, <deitado>, <de-pé>, <esquerda>, <bloco>, <sobre>}

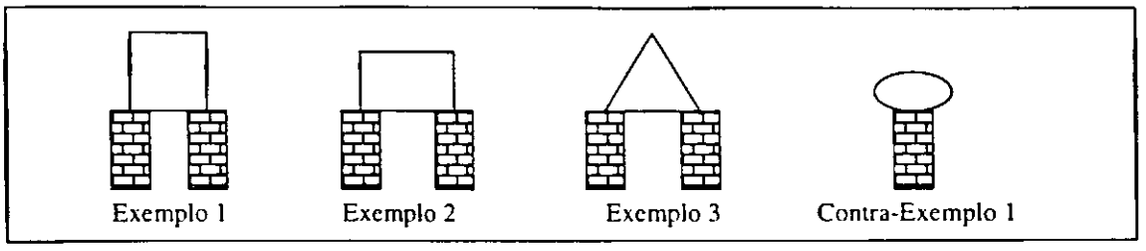
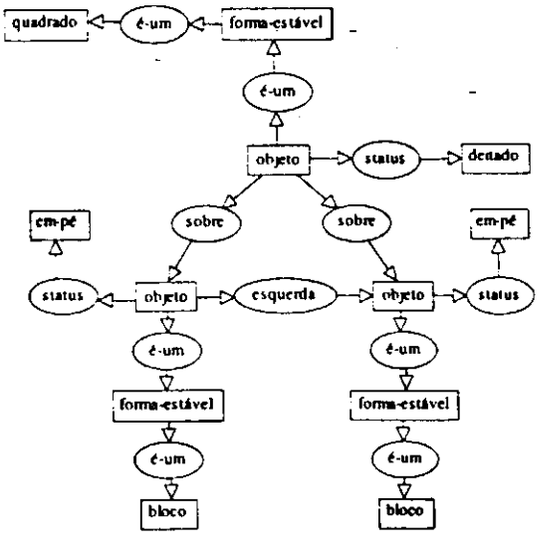
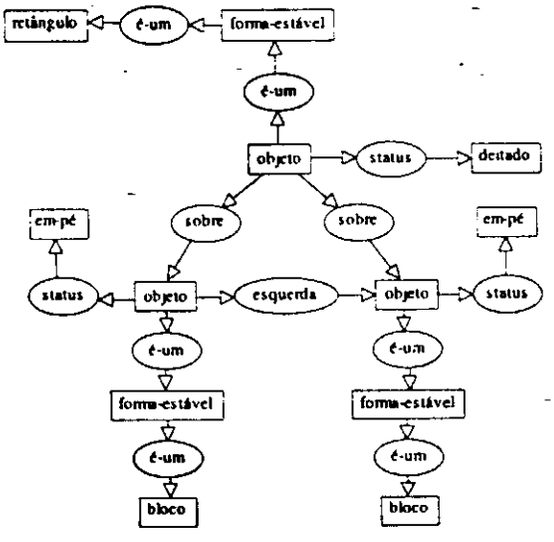


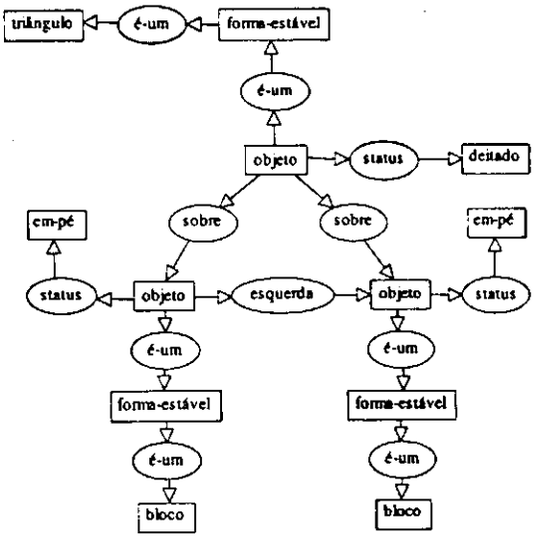
Fig. 2.9 - Amostra formada por 3 exemplos e 1 contra-exemplos de arco.



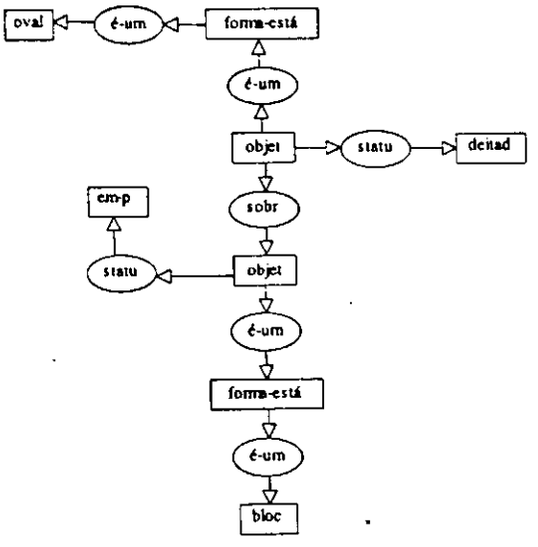
Ex1



Ex2



Ex3



Cex1

Fig. 2.10 - Representação da amostra de arco em grafos conceituais.

Os conceitos quadrado, oval, triângulo, retângulo e forma-instável não são selecionadas pois só aparecem uma única vez. A partir desse primeiro conjunto, são construídas classes mais específicas. Assim, a classe composta de caminhos de comprimento 1 (E_1) é definida pela concatenação dos caminhos que compõem E_0. Considerando sempre o critério de validade, tem-se:

E_1 = {<é-um,forma-estável>, <é-um,bloco>, <forma-estável,é-um> ,<objeto,é-um>, <objeto,status>, <objeto,sobre>, <objeto,esquerda>, <status,de-pé>, <status,deitado>,<esquerda,objeto>, <sobre,objeto>}

A partir de E_1, construímos a classe de equivalência E_{i+1}= E_2.

E_2 = {<é-um,forma-estável,é-um>, <forma-estável,é-um,bloco>, <objeto,é-um,forma-estável>, <objeto,status,de-pé>, <objeto,status,deitado>, <objeto,sobre,objeto>, <objeto,esquerda,objeto>, <esquerda,objeto,é-um>, <esquerda,objeto,status>, <sobre,objeto,é-um>, <sobre,objeto,status>, <sobre,objeto,esquerda>}

Esse processo continua até que E_n seja vazio. No final retiram-se de E_i os caminhos que não iniciem por conceitos e são retidos apenas os caminhos de comprimento i a partir dos quais não se gerou nenhum outro caminho de comprimento i+2.

O resultado final obtido por esse método é:

E_n = {<objeto,é-um,forma,estável>,<objeto,status,deitado>,<objeto,sobre,objeto>, <forma-estável,é-um,bloco>,<objeto,estatus,de-pé>,<objeto,esquerda,objeto>, <objeto,sobre,objeto,é-um,forma-estável>,<objeto,é-um,forma-estável,é-um,bloco>, <objeto,sobre,objeto,status,de-pé>,<objeto,sobre,objeto,esquerda,objeto>, <objeto,esquerda,objeto,status,de-pé>,<objeto,esquerda,objeto,é-um,forma-estável>, <objeto,esquerda,objeto,é-um,forma-estável,é-um,bloco>, <objeto,sobre,objeto,esquerda,objeto,é-um,forma-estável>, <objeto,sobre,objeto,esquerda,objeto,é-um,forma-estável>, <objeto,sobre,objeto,é-um,forma-estável,é-um,bloco>, <objeto,sobre,objeto,esquerda,objeto,é-um,forma-estável,é-um,bloco>}.

2.6.4.2 Assimilação e Discriminação

A partir dessas similaridades, INNE reúne esses caminhos em dois tipos de conjuntos: os de assimilação e os de discriminação; o primeiro permite identificar todos os elementos do conceito, enquanto que o segundo admite apenas aqueles que não possuem ruído. O conceito aprendido é então definido pela observação desse dois conjuntos. Para o exemplo acima, o INNE gera os seguintes conjuntos:

- Assimilação:

{<objeto,status,deitado> &

<objeto,sobre,objeto,status,de-pé> &
<objeto,sobre,objeto,esquerda,objeto,é-um,forma-estável>}

- Discriminação:

{<objeto,status,deitado> &
<objeto,sobre,objeto,status,de-pé,objeto,sobre,objeto,status,de-pé> &
<objeto,sobre,objeto,esquerda,objeto,é-um,forma-estável> &
<objeto,sobre,objeto,esquerda,objeto,é-um,forma-estável> &
<objeto,sobre,objeto,esquerda,objeto,é-um,forma-estável> &
<objeto,sobre,objeto,é-um,forma-estável,é-um,bloco>}

2.7 Resumo do Capítulo

Apresentamos neste capítulo conceitos relacionados à aprendizagem de máquina, focalizando nossa abordagem na aprendizagem a partir de exemplos. Discutimos sobre os aspectos que diferenciam os dois principais representantes deste tipo de aprendizagem; a Aprendizagem Baseada em Similaridades (SBL) e a Aprendizagem Baseada em Explicação (EBL). Vimos também os princípios que uma linguagem de descrição deve adotar para suportar os modelos de classes e objetos para sistemas de aprendizagem deste tipo. Apresentamos ainda alguns métodos de aprendizagem (alguns deles clássicos), descrevendo suas principais características, propriedades, vantagens e inconvenientes.

Procurou-se neste capítulo incluir os conceitos que, juntamente com as do capítulo anterior, fossem suficientes para que em se apresentando um programa de aprendizagem de máquina, pudéssemos analisá-lo e situá-lo dentro da IA e mais especificamente, dentro da Aprendizagem de Máquina, como estabelecido inicialmente na Introdução.

O próximo capítulo apresenta o projeto de concepção e desenvolvimento do Agente Racional SAID, onde se insere o nosso trabalho.

Capítulo 3

O Agente Racional SAID

3.1 Introdução

O agente racional SAID (Somente Abduzir, Induzir, Deduzir) é um sistema inteligente concebido com a capacidade de construir seu próprio conhecimento através da interação com um agente humano. Esse sistema é capaz de fornecer explicações sobre suas decisões e pode ser visto como um sistema de aquisição de conhecimento organizado em três níveis hierárquicos:

1 Um sistema de representação de conhecimento que provê os meios necessários à construção de raciocínios sobre conhecimentos incompletos e passíveis de erro (as Teorias Semi-Empíricas);

2 Uma metodologia de integração de diversos mecanismos de aprendizagem para a geração do conhecimento; e

3 Um ambiente multi-agente que provê os elementos necessários para o controle por diálogo entre os diversos agentes envolvidos na geração e evolução do conhecimento (o protocolo de comunicação MOSCA: Mestre + Oráculo + Sonda + Cliente + Aprendiz [REITZ 1992]).

Na concepção do agente racional, pressupõe-se que o conhecimento é produto da interação de três componentes:

1 A *linguagem*, cujas sentenças expressam os exemplos dos conceitos a aprender e a partir das quais os fatos são as regularidades observadas;

2 Os *objetos*, que expressam as restrições sobre os fatos;

3 As *conjecturas*, que expressam as restrições sobre os objetos.

A partir dos fatos, o agente racional constrói hipóteses a respeito de objetos e conjecturas a respeito das relações entre objetos que, quando aceitas por um especialista do domínio, são denominadas *lemas* e *provas*, respectivamente. O conjunto de lemas e provas é o conhecimento obtido por um agente racional por repetições do ciclo de raciocínio abdução, indução, dedução e argumentação. O controle da evolução do conhecimento de um agente racional se dá por processos de

crítica baseados na Lógica das Provas e Refutações [LAKATOS 1976], realizados pelos componentes do protocolo de aprendizagem MOSCA. Cada etapa do processo de construção do conhecimento pelo agente racional (*Aprendiz*), é avaliada por um dos outros agentes do protocolo, de acordo com as seguintes funções:

- O Oráculo: avalia a qualidade de uma indução;
- O Mestre: avalia a qualidade de uma abdução;
- A Sonda: avalia a qualidade de uma dedução;
- O Cliente: avalia a qualidade de uma argumentação.

Considerando-se o exposto acima, podemos utilizar neste trabalho a seguinte definição de agente racional:

> "Agente racional é um sistema que, para um conjunto de exemplos e de objeções, constrói as provas empíricas e/ou analógicas para uma conjectura e determina seu grau de validade" [SALLANTIN 1990].

Na figura 3.1 é apresentado um esquema genérico do agente racional SAID.

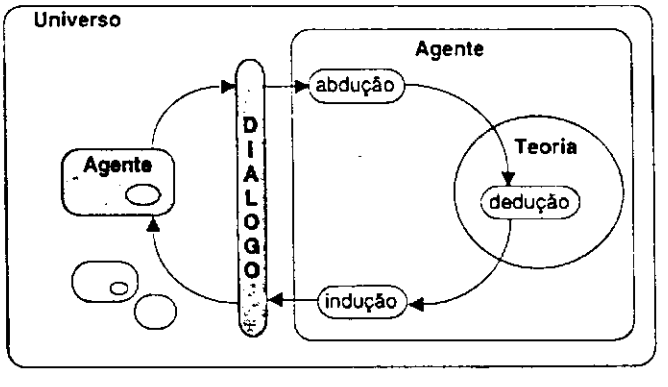


Fig. 3.1- Agente Racional SAID e seus mecanismos de inferência

3.2 Aquisição do Conhecimento e a Lógica da Descoberta Científica

Em sua Lógica da Descoberta Científica, Popper [POPPER 1993] estabelece que uma teoria científica não é obtida por indução a partir de um conjunto de fatos observados, mas pela conjuntura de invenção de hipóteses, conjecturas e até mesmo por "adivinhações". Segundo ele, uma teoria científica descreve coisas existentes em um mundo real e que, para a Ciência, não importa como esta é obtida mas sim como se verifica o seu grau de verossimilhança com o real. Para que uma teoria seja considerada científica, ela deve satisfazer o princípio da refutabilidade; ou seja, ela deve arriscar-se a fazer predições passíveis de falsificação. Uma teoria é *falsificável* quando existe um con-

junto potencial de falsificadores que possa ser submetido a testes no mundo real. Uma teoria é mais falsificável que outra na medida que esse conjunto de falsificadores é maior ou menor. Uma teoria científica deve ser definida de forma que possa predizer algo sobre o mundo real.

3.2.1 A Lógica das Conjecturas e Refutações

Popper mostrou que não seria necessário se ter como princípio científico a lógica da indução; a Ciência teria como fundamento uma *Lógica das Conjecturas e Refutações*. Popper afirma que a Ciência progride através da lógica situacional de resolução de problemas [POPPER 1975A]. O que ele chama *lógica situacional de problemas*, ou *análise situacional*, está relacionado a uma explicação conjectural ou experimental de alguma ação humana que recorra à situação em que o agente se encontra. Nesse sentido, o conhecimento científico se desenvolve numa cadeia de problemas e soluções de problemas, como mostrado na figura 3.2

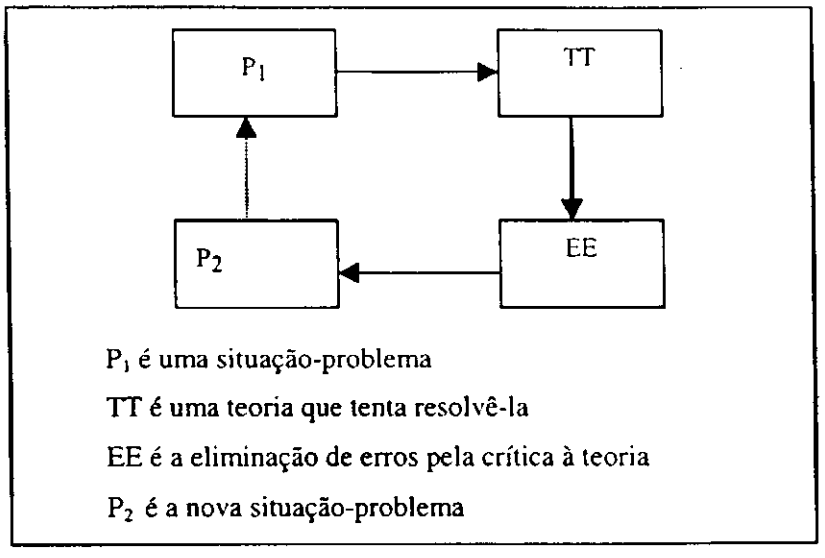


Fig. 3.2 - Modelo geral para Solução de Problemas

Neste esquema geral de solução de problemas, várias TTs competem para ser uma solução para a situação-problema em questão, e as conjecturas e refutações podem ser identificadas com os mecanismos dessa abordagem de variação e seleção. O mecanismo de variação é, na construção de uma teoria científica, representado pela fase de geração de hipóteses e o mecanismo de seleção é representado pela fase de testes da teoria em situações que ocorrem no mundo real.

3.2.2 A Lógica das Provas e Refutações

Seguindo os passos de Popper, Lakatos [LAKATOS 1976] formulou sua Lógica da Descoberta Matemática baseado no princípio da refutabilidade. Segundo ele, seu objetivo ao formular a sua Lógica

das Provas e Refutações baseia-se na constatação de que:

> "... a matemática não formal, semi-empírica, não progride mediante monótono aumento do número de teoremas indubitavelmente estabelecidos, mas mediante incessante aperfeiçoamento de opiniões por especulação e crítica, pela lógica das provas e refutações." [LAKATOS 1976].

A Lógica das Provas e Refutações parte do princípio de que o conhecimento na Matemática informal é hipotético, conjectural, e desenvolvido por meio de especulações e críticas. O seu objetivo é estudar a construção de uma prova e determinar o seu domínio de validade pela análise de exemplos e contra-exemplos dessa prova. Uma prova é um conjunto de lemas determinados a partir da análise de uma conjectura e suas sub-conjecturas. A análise de uma prova é realizada pela apresentação de exemplos e contra-exemplos. É desta forma que lemas são modificados e lemas são descobertos, em função das críticas ou refutações a que as provas são submetidas.

Na figura 3.3 é mostrado um modelo simplificado de Lakatos para a heurística da descoberta matemática.

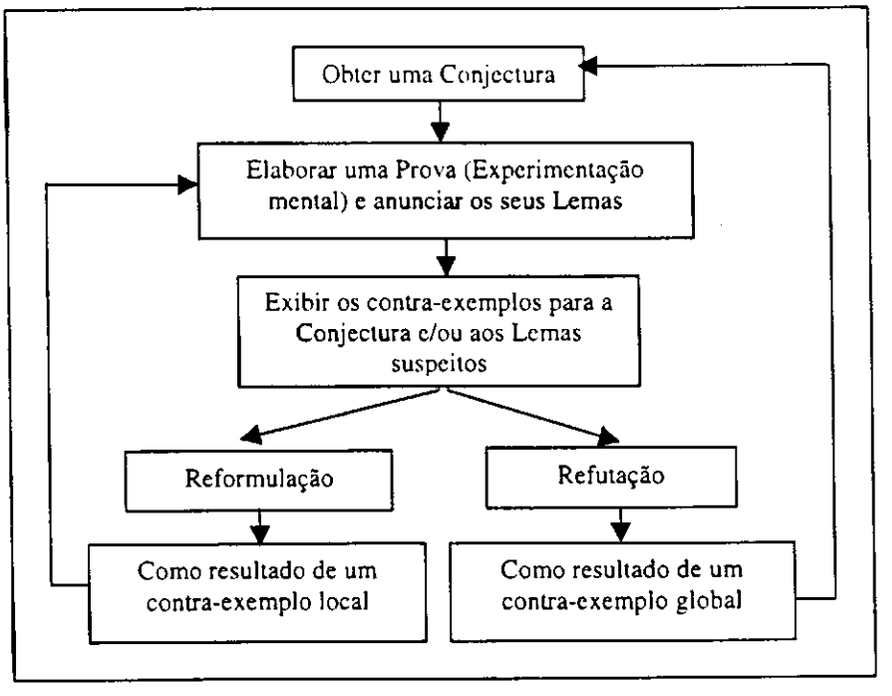


Fig. 3.3 - Modelo simplificado da Lógica das Provas e Refutações

A Lógica das Provas e Refutações pode ser sucintamente descrita pelos seguintes passos:

1 Obter uma conjectura;

2 Elaborar uma prova e enunciar os lemas que constitui a prova analítica inicial;

3 Exibir os contra-exemplos para a conjectura e/ou para lemas suspeitos;

4 Construir a prova analítica para retificação de lemas ou para adicionar lemas, que devem ser também introduzidos como condição à conjectura.

Pode-se notar que o método das provas e refutações exige uma abordagem interativa. Este método se interessa tanto pela elaboração de uma prova da conjectura como pela crítica dessa prova.

3.3 O SAID como Sistema de Apoio à Descoberta Científica

Considere um sistema de aprendizagem dotado das seguintes características:

- Apto a interpretar estruturas simbólicas;
- Consciente de suas limitações;
- Age em acordo lógico com suas crenças;
- Capaz de adaptar suas ações a mudanças em seu conhecimento.

Um sistema como tal é capaz de melhorar suas representações do mundo externo e de melhor interagir com esse mundo. Essa capacidade de construir e evoluir suas representações do mundo pode ser adicionada à aptidão de aprender de um agente inteligente.

Trabalhos realizados por diversos autores [SALLANTIN 1985; KONOLIGE 1985; NILSON 1986; BRATMAN 1986; MÜLLER 1987; FERNEDA 1992B], mostram a possibilidade de se construir estes sistemas. A arquitetura de um agente racional mostrada na Fig. 3.1, reflete a sua condição de sistema de apoio à descoberta científica. Este fato evidencia-se pela utilização dos mecanismos de inferência (abdução, indução e dedução), discutidos mais adiante (seção 3.5), como metodologia empregada na construção de seu conhecimento. Como esta arquitetura utiliza o princípio da refutabilidade como critério de validação do seu conhecimento, pode ser caracterizado como um sistema popperiano [SALLANTIN 1997].

3.4 Elementos de um Sistema de Apoio à Descoberta Científica

3.4.1 Teorias Semi-Empíricas (TSE)

O modelo de lógica da descoberta científica adotado pela TSE tem o objetivo de estudar a prova de uma conjectura visando determinar o que intervém no exame da validade de uma prova analisando o seu poder de predição, e o que intervém no exame da relevância de uma prova analisando o seu poder de explicação. TSE é uma forma de representação do conhecimento que expressa uma con-

jectura através de relacionamentos e restrições sobre relacionamentos entre sentenças de uma linguagem [SALLANTIN 1991A].

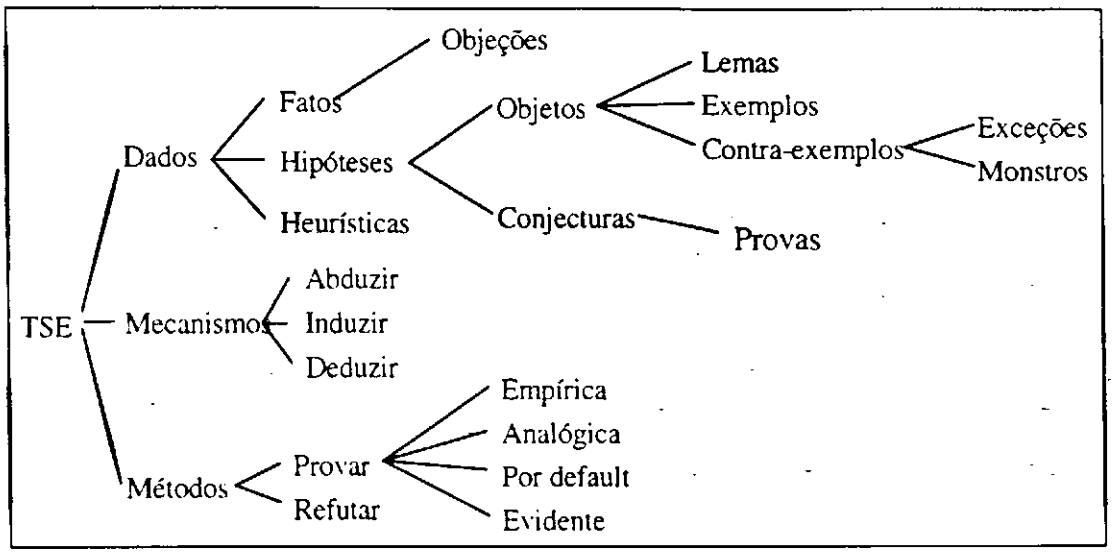


Fig. 3.4 - Taxonomia dos termos usados na TSE

A Figura 3.4 descreve uma taxonomia dos termos utilizados para expressar o conhecimento na TSE. Esta taxonomia é baseada no trabalho de Addis [ADDIS 1988], que revisou o trabalho de Charles S. Peirce sobre modelagem do conhecimento e foi complementada por Sallantin [SALLANTIN 1991A] com os termos utilizados na Lógica das Provas e Refutações.

Na TSE os conhecimentos são classificados como:

- **Dados**. Representam o conhecimento de um agente racional. Esse conhecimento é representado por expressões a serem geradas e evoluídas. O conhecimento é capturado em uma das três formas seguintes:
  - **Fatos**. São sentenças que expressam regularidades, para as quais um grau de validade pode ser atribuído.
  - **Hipóteses**. São sentenças para as quais um grau de relevância pode ser atribuído.
  - **Heurísticas**. Definem as formas que os fatos e hipóteses podem assumir.
- Mecanismos para a geração de conhecimento.
  - **Abdução**. Descobre novas regularidades, para a organização desse conhecimento.
  - **Indução**. Sugere novas hipóteses e novas heurísticas e realiza a propagação de restrições sobre eles
  - **Dedução**. Determina as conseqüências lógicas do conhecimento.

- **Métodos relacionados às interações com um agente externo:** Para criticar ou propor uma sentença a ser provada. Estes métodos examinam a adequação de conhecimentos tais como ser um lema, ser uma objeção, ser uma prova e ser uma conjectura.

Um conhecimento na TSE é julgado por uma medida de *validade* e de *pertinência* de suas deduções. A validade se avalia em função do número de resultados justos e a pertinência, em função do número de resultados provados e aceitos pelo usuário.

3.4.2 Um protocolo para a aprendizagem

No protocolo de aprendizagem MOSCA [REITZ 1992], cujo ambiente é apresentado na figura 3.5, podemos destacar essencialmente cinco componentes distintos:

- **Aprendiz**: Constrói uma hipótese aprendida a partir da amostra de exemplos e contra-exemplos disponibilizada previamente;
- **Oráculo**: Produz soluções irrefutáveis de problemas, na forma de pares <*problema*, *solução*>;
- **Cliente**: Submete problemas ao aprendiz e espera soluções;
- **Sonda**: Produz soluções refutáveis na forma de pares <*problema,solução*>, obrigando o aprendiz a produzir argumentações; e
- **Mestre**: Analisa as argumentações do aprendiz e as critica.

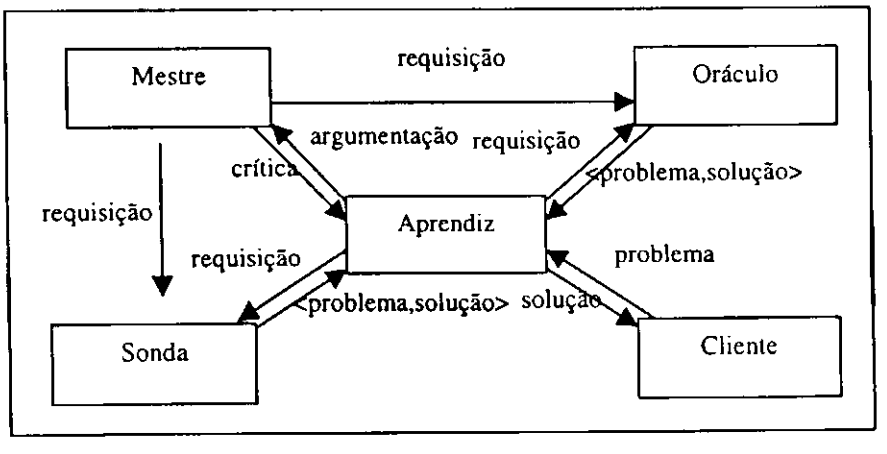


Fig. 3.5 - Protocolo de aprendizagem MOSCA

Para este protocolo, temos a seguinte análise funcional:

- O cliente submete um problema ao aprendiz e espera dele uma solução;
- O aprendiz requisita pares <*problema,solução*> ao oráculo;

- O aprendiz pode requisitar ao oráculo um ou mais problemas resolvidos. Essa requisição pode ser feita de duas maneiras:

  - quando a escolha do problema é deixado a critério do oráculo (aleatoriamente ou segundo um plano de ensino predeterminado), a requisição é feita simplesmente pelo envio de um sinal;

  - quando o aprendiz deseja aprender a resolver um certo problema, ele envia ao oráculo uma requisição de objetos do problema que lhe interessa (isto, é claro, deve ser feito de forma controlada, senão o aprendiz se contenta apenas em questionar o oráculo, sem nada aprender);

- O oráculo envia os pares, que são então armazenados pelo aprendiz e compõem a amostra a ser usada para a aprendizagem. Essa amostra pode eventualmente ser modificada, visto que pode estar sujeita a erros ou simplesmente se mostrar não apropriada para um certo contexto. Toda alteração na amostra faz com que o aprendiz reavalie a hipótese aprendida. Essa hipótese é extraída de um espaço de hipóteses e satisfaz um critério de aprendizagem. É claro que qualquer hipótese aprendida reconhecerá os componentes da amostra como exemplos e contra-exemplos da mesma forma como apresentados pelo oráculo;

- O aprendiz também recebe problemas resolvidos da sonda. No entanto, essas soluções podem ser intencionalmente errôneas. Para cada problema resolvido apresentado pela sonda, o aprendiz compara a solução com a sua própria e apresenta ao mestre uma argumentação justificando sua solução. Essas argumentações podem ser de dois tipos: *explicações* (caso as duas soluções coincidam) ou *objeções* (caso contrário);

- Para cada argumentação apresentada, o aprendiz recebe do mestre uma crítica. Sempre que possível, cada crítica negativa faz com que o aprendiz apresente uma argumentação alternativa. Quando nenhuma argumentação alternativa é possível ao aprendiz, ele dispõe das seguintes alternativas:

  - Excluir este problema não argumentável (classificando-o como *monstro* ou como *contra-exemplo*, podendo no segundo caso ser este incluído como tal na amostra para uma possível reaprendizagem),

  - Aceitá-lo, mesmo assim, como objeto do conceito sendo aprendido (classificando-o como *exceção*),

  - Reformular sua hipótese aprendida através de uma nova execução do processo de aprendizagem, incluindo o problema não argumentável como um novo elemento da amostra (classificando-o como *exemplo*), ou ainda

- Enfraquecer a confiança na hipótese aprendida (neste caso, presume-se um limite mínimo para esta confiança que, se ultrapassada, deve acarretar uma reaprendizagem na qual as críticas acumuladas devem influenciar a construção de uma nova hipótese aprendida).

- mestre, enviando um sinal ou uma requisição de um problema específico, controla a produção de problemas resolvidos da sonda.

3.4.3 A estrutura do Aprendiz

No protocolo MOSCA, a forma dos problemas que o aprendiz pode resolver se restringe à identificação de conceitos. O aprendiz, baseado na aprendizagem a partir de exemplos e contra-exemplos, formula uma conjectura para ser validada, revisada, ou utilizada, onde:

- O problema é definido através de um objeto e do conceito que este objeto representa,
- A solução para um Problema consiste na formulação de uma crença na relação de pertinência de um objeto a um conceito e
- As conjecturas são formadas a partir de combinações lógicas de componentes elementares (*regularidades*). O aprendiz dispõe de mecanismos para a construção de uma conjectura.

3.4.3.1 Objetivo do Aprendiz

O objetivo do aprendiz é construir uma conjectura sobre um conceito. Assume-se que o aprendiz não recebe nenhuma outra informação senão uma amostra composta de exemplos e contra-exemplos deste conceito, oferecidas pelo oráculo, e constrói, observando regularidades na amostra, regras (i.e., conjecturas) na forma:

Se o objeto satisfaz a regularidade R, então sentença S tem crença C

3.4.3.2 Princípios para a aplicação de uma conjectura

A aplicação do conjunto de regras aprendidas se dá de uma forma simples: quando a sonda ou o oráculo apresentam um problema resolvido ao aprendiz, este determina uma crença da pertinência de um objeto ao conceito em dois passos:

1. A aplicação do conjunto de regras que formam a conjectura, resultando em conjunto de crenças, uma para cada regra aplicada.
2. A agregação dos valores de crença em uma única crença, que será associada ao problema

apresentado (um par <*objeto,conceito*>).

Essa agregação pode ser realizada de várias formas. As mais usadas são:

- Abordagem da ***Lógica Clássica***: se o conjunto de valores de crença é {*verdadeiro, falso*}, então a agregação pode ser feita tanto por conjunção (se alguma regra gera o valor *falso*, a agregação também o será) como por disjunção (se alguma regra gera o valor *verdadeiro*, a agregação também o será).
- Abordagem da ***Lógica Majoritária***: Dois limites permitem regular a natureza mais conjuntiva ou mais disjuntiva da agregação das regras aprendidas. O primeiro limite define o número mínimo de regras que devem ser verdadeiras para que a agregação também seja verdadeira. O segundo limite define o número mínimo de regras que devem ser falsas para que a agregação também seja falsa. Como estes limites não são necessariamente complementares, há casos em que o resultado não é nem *falso* nem *verdadeiro*: é *silêncio*. Esta segunda abordagem engloba, evidentemente, a primeira. A definição de Lógica Majoritária será vista mais adiante, em [3.4.6].

3.4.3.3 Princípios de construção de uma argumentação

Após serem mostrados os princípios de construção de uma conjectura, presentes em todos os trabalhos sobre aprendizagem automática, serão expostos mecanismos que permitem uma revisão do conhecimento aprendido. No protocolo proposto, essa revisão é empreendida em dois casos exclusivos:

- Um novo exemplo/contra-exemplo é proposto pelo oráculo, o qual vem enriquecer a amostra do aprendiz, o que questiona as regularidades observadas (aparição de novas regularidades, desaparecimento de antigas), e portanto a conjectura aprendida.
- A argumentação produzida pelo aprendiz é criticada, o que, indiretamente, questiona essa mesma conjectura.

O primeiro caso de revisão consiste em estudar a natureza incremental dos mecanismos de detecção de regularidades na amostra.

Antes da verificação dos efeitos de uma crítica a uma argumentação sobre o conhecimento aprendido, deve-se examinar como tal argumentação pode ser construída. Distingue-se duas formas de argumentação:

- **Explicação**: o aprendiz explica quando a crença por ele calculada está em conformidade

com a crença dada pela sonda, ou mais simplesmente, quando ele busca uma justificativa para a crença transmitida a um cliente que lhe submeteu um objeto. No âmbito da TSE, uma explicação é formada por um conjunto minimal de regras que bastam à obtenção do resultado. Esse conjunto não é necessariamente único, e serão vistas, na próxima seção, técnicas que permitem a resolução de conflitos. Por exemplo, se a crença final proposta é verdadeira, sabendo-se que dez regras sobre quinze são verdadeiras, quando somente oito teriam sido suficientes, o aprendiz deve escolher oito dessas regras sobre as dez verdadeiras. As regras retidas são então propostas como uma explicação da crença obtida.

- **Objeção**: o aprendiz gera uma objeção quando uma crença que ele calcula é diferente da proposta pela sonda. No âmbito da TSE, uma objeção [BARBOUX 1990] é formada pelo conjunto das regras que bastam para explicar o fracasso na objeção da crença proposta pela sonda. Como para as explicações, esse conjunto não é necessariamente único (alguns métodos de decisão serão apresentados na próxima seção).

3.4.3.4 Princípios de exploração de uma crítica

Quando o aprendiz propõe sua argumentação ao mestre, este responde através de uma crítica. A princípio, imagina-se que essa crítica seja binária: aprovação ou desaprovação; diz-se que o aprendiz é criticado quando ele recebe uma crítica negativa. Se o aprendiz pode propor uma argumentação, no caso em que ele dispõe de vários conjuntos de regras candidatas à construção de uma argumentação, isso é feito. No melhor dos casos, ele encontrará uma argumentação que não será criticada. Senão, ele deve mudar seu modo de argumentação, o que, no caso das TSE, consiste em reformular a conjectura, excluindo o objeto do seu domínio de definição; ou seja, a resposta sugerida pelo aprendiz neste caso será o silêncio. Isso significa que o aprendiz, antes de utilizar uma conjectura, deve verificar se o objeto considerado não faz parte das exceções aceitas para o mesmo.

Quando o aprendiz é criticado sobre uma argumentação, é possível ao mesmo classificar as regras aprendidas das seguintes formas:

- algumas regras não foram ainda envolvidas na argumentação;
- outras regras foram utilizadas para construir argumentações, as quais foram criticadas ou não.

Denomina-se *lema* uma regra que já tenha sido utilizada em argumentações e que não receberam ainda críticas negativas. Esses lemas são argumentos importantes, pois o mestre sempre reconheceu seu papel explicativo. Na realidade, eles são muito mais que isso: eles formam termos de uma linguagem que foi admitida pelas duas partes, o aprendiz e o mestre, e serve de suporte ao di-

álogo entre ambos.

Essa divisão das regras em lemas e não-lemas sugere que uma estratégia para determinar o melhor conjunto de regras para a construção de uma argumentação pode seguir dois esquemas opostos:

- se o aprendiz está pronto a assumir riscos, então ele escolherá preferencialmente uma argumentação composta de um número máximo de lemas. Uma crítica negativa por parte do mestre, corresponde a um questionamento da linguagem do aprendiz.
- senão, o aprendiz não se arrisca, e escolhe preferencialmente regras que não são lemas. Aqui, ele busca antes de mais nada preservar sua linguagem de comunicação, conservando o que ele já construiu.

Essa estratégia de exploração das críticas pode tomar as mais diversas formas: é possível imaginar que a cada regra pode associar-se um contador de críticas negativas e quanto mais esse contador cresce, mais a qualidade dessa regra é contestada. Esse contador pode ser utilizado para selecionar um conjunto de regras que será utilizado para a construção de uma argumentação.

A gestão da argumentação gera um certo número de problemas que são mostrados a seguir:

1 a revisão de uma hipótese aprendida deve questionar as informações sobre as críticas recebidas contidas na função de argumentação;

2 a exaustão das argumentações possíveis, quando o aprendiz é sempre criticado, deve conduzir à revisão de sua conjectura, mesmo quando ela está em perfeita adequação com os dados da amostra;

3 uma argumentação criticada questiona mais um modo de argumentação do que os argumentos que a compõem.

3.4.4 O esquema mental

TSE é uma teoria que contém seu próprio mecanismo de evolução. Esse mecanismo é baseado na idéia de que uma linguagem é um conjunto ordenado de sentenças e que um sistema de crenças é um conjunto ordenado de valores de verdade. Quanto a isso, Newell [NEWELL 1982] mostrou que conhecimento pode ser entendido tão independentemente quanto possível de qualquer linguagem de representação. Em outras palavras, pode-se abstrair conhecimento de sua representação. Para se chegar a uma abstração, um agente racional deve ser capaz de associar objetos a sentenças e a outros objetos.

A linguagem formada pelo conjunto de sentenças é usada tanto para capturar como para comunicar conhecimento (sem essa comunicação através de uma linguagem, seria muito difícil para um agente realizar as interpretações das sentenças, pois não poderia receber formulações de críticas por outros agentes!).

Visando a definição de objetos como interpretações de um conjunto de sentenças e de conjecturas como interpretações dos conjuntos de objetos, apresenta-se a noção de esquema mental [AUBERT 1990; SALLANTIN 1991B].

Cada agente racional é composto de uma TSE (representando o conhecimento) e um esquema mental (que permite a evolução do conhecimento pela aprendizagem e pela revisão do que foi aprendido).

Um esquema mental é uma tripla (L,C,Δ), onde:

- L é um conjunto de sentenças ordenado pela relação de ordem parcial $\leq_L$,
- C é um conjunto de valores de crenças ordenado pela relação de ordem parcial $\leq_C$, formando um reticulado, tendo portanto um máximo (**1**) e um mínimo (**?**).
- $\Delta : L \times L \rightarrow C$ é uma função de crença, definida como: se s_1 e s_2 são duas sentenças e $c = \Delta(s_1,s_2)$, então a sentença s_1 explica (com uma crença c) a sentença s_2 no esquema mental.

Maiores detalhes sobre o esquema mental utilizado pelo SAID estão fora do escopo desta abordagem, podendo ser encontradas nos trabalhos de Sallantin [SALLANTIN 1991A,1991B].

3.4.5 Um sistema de crenças

O SAID adota um sistema de crenças baseado na Lógica Multivalorada de M. Ginsberg [GINSBERG 1988, 1990]. Uma *crença* é um elemento do conjunto C, parcialmente ordenado por dois tipos de relação:

- $\leq_k$: a respeito desta relação, C é um reticulado com *silêncio* como seu mínimo e $\perp$ como seu máximo.
- $\leq_t$: a respeito desta relação, C é um reticulado cujo máximo é *verdadeiro* e cujo mínimo é *falso*.

A relação $\leq_k$ é normalmente interpretada por *é menos conhecida que* e $\leq_t$ por *é menos verdadeiro que*. A crença $\perp$, representando uma contradição, é tal que é *verdadeiro* $\leq_k \perp$ e é *falso* $\leq_t \perp$.

Um conjunto de crenças é associado a cada um dos papéis envolvidos no protocolo de aprendizagem MOSCA. O conjunto de todas as crenças neste protocolo é $C = C_M \cup C_O \cup C_S \cup C_C \cup C_A$, onde:

- C_M = {objetado, não-objetado, silêncio}
- C_O = {verdadeiro-O, falso-O}
- C_S = {verdadeiro-S, falso-S}
- C_C = { } e
- C_A = {aceito, contestado, silêncio, ⊥ (contradição)}.

Na Figura 3.6 é apresentado o reticulado de crenças, isto é, a ordem entre os valores de crença. A figura pode ser interpretada da seguintes maneira:

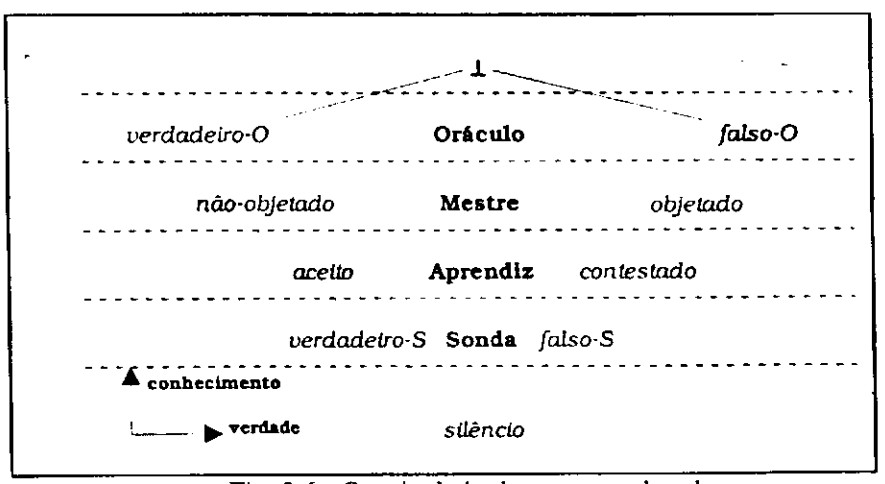


Fig. 3.6 - O reticulado de crenças adotado

1 Ordem entre as crenças de conhecimento:

$$silêncio \leq_k verdadeiro\text{-}S \leq_k aceito \leq_k não\text{-}objetado \leq_k verdadeiro\text{-}O \leq_k \perp$$

e

$$silêncio \leq_k falso\text{-}S \leq_k contestado \leq_k objetado \leq_k falso\text{-}O \leq_k \perp$$

2 Ordem entre as crenças de verdade:

$$falso\text{-}O \leq_t objetado \leq_t contestado \leq_t falso\text{-}S \leq_t silêncio$$

e

$$verdadeiro\text{-}S \leq_t aceito \leq_t não\text{-}objetado \leq_t verdadeiro\text{-}O$$

3.4.6 Mecanismos para a Geração e Evolução do Conhecimento

Os mecanismos da TSE são responsáveis pela geração e pela evolução do conhecimento (i.e., dos esquemas mentais). Como vimos anteriormente, existem na lógica clássica, três mecanismos responsáveis pela construção e evolução de uma teoria: *abdução*, *indução* e *dedução* [SALLANTIN 1991A, 1991B].

Abdução associa uma crença a uma proposição sempre que esta proposição não pode ser deduzida da teoria mas mantém esta teoria coerente, e vale como uma conseqüência desta proposição. A abdução utiliza a *Lógica Majoritária*.

Em um raciocínio baseado na lógica clássica, o valor-verdade de uma sentença conjuntiva só é verdadeiro quando todos os valores-verdade associados a cada um de seus átomos elementares também são verdadeiros. Num raciocínio empírico, uma sentença conjuntiva pode ser aceita como verdadeira quando uma certa quantidade de seus átomos elementares componentes forem avaliados como verdadeiros. Ou seja, dado um conjunto de sentenças $S = \{s_1,...,s_n\}$, $\mathrm{MAJ}_\Phi S$ (lei majoritária com limite de aceitação Φ) indica que a conjunção formada pelo conjunto de sentenças S é avaliada como verdadeira quando $|S_+|/|S| \geq \Phi$. A isto denomina-se Lógica Majoritária.

Indução gera uma nova regra relacionando proposições.

Dedução utiliza regras para inferir uma nova crença sobre uma proposição sempre que crenças sobre suas condições já estejam estabelecidas.

Trabalhos anteriores [BARBOUX 1990; LIQUIÉRE 1990A, 1990B, 1990C, 1992; MEPHU-NGUIFU 1993A, 1993B, 1993C] se restringiram à linguagem da lógica proposicional. Ferneda [FERNEDA 1992B], adotou a lógica de predicados de primeira ordem, visto seu interesse em construir conjecturas sobre objetos estruturados (ou complexos). Regras geradas por indução terão a forma $\rightarrow \{s_1,...,s_n\}$, significando a existência de uma dependência entre o conjunto de proposições s_1, ..., s_n. Esta dependência é definida através de um conjunto de unificações entre as variáveis presentes na proposição.

A semântica da lei de agregação é dada por:

$$\Delta(x,\mathrm{MAJ}_\Phi\{e_1,...,e_n\}) \leq_C C, \text{ onde } |\{e_i \mid \Delta(x,e_i) \leq_C C\}| \geq \Phi$$

e

$$\Delta(x,e_2) \geq_C min_C(\Delta(x,e_1),\Delta(x,\rightarrow\{e_1,e_2\}))$$

Pode-se ver como os três mecanismos atuam:

$$\frac{(\text{Abdução})\ \Delta(\alpha, \text{MAJ}_{\Phi}\{r_1,\ldots,r_n\}) \qquad (\text{Indução})\ \Delta(\alpha, \rightarrow\{\text{MAJ}_{\Phi}\{r_1,\ldots,r_n\}, K\})}{(\text{Dedução})\ \Delta(\alpha, K)}$$

Uma ordem parcial $\leq K$ sobre o conjunto de esquemas mentais é definido como:

$$\Delta \leq K \Delta' \overset{\text{def}}{\Leftrightarrow} \forall x,y \in L, \Delta(x,y) \leq K \Delta'(x,y)$$

T. S. Kuhn [KUHN 1976] viu um ciclo no processo de construção do conhecimento. Este ciclo é composto de três fases: evolução normal, crise, e revolução, como mostra a Fig. 3.7.

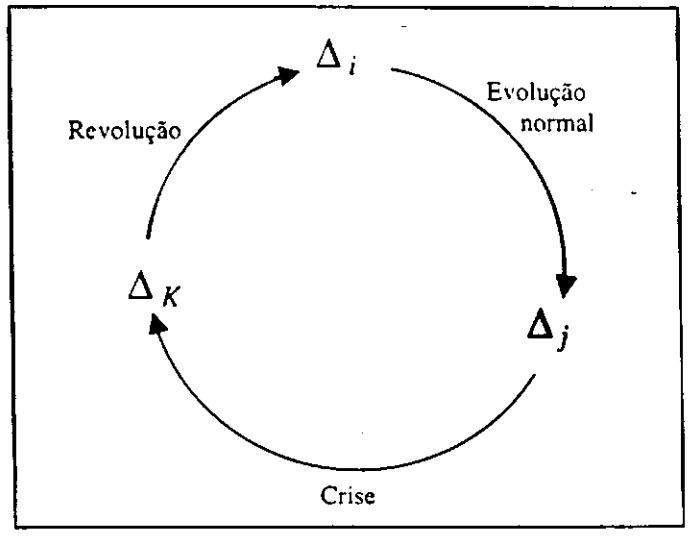


Fig. 3.7 - Ciclo de evolução do conhecimento segundo T.S.Kuhn [KUHN 1983]

A fase de evolução normal corresponde à aplicação das técnicas de evolução de conhecimento da TSE. Nesta fase a evolução pode ser vista como uma seqüência $\Delta_i \leq_k \Delta_{i+1} \leq_k \ldots \leq_k \Delta_j$. Sejam Δ_{s_1} e Δ_{s_2} esquemas mentais. Diz-se que Δ_{s_1} evolui para Δ_{s_2} (ou seja, $\Delta_{s_1} \mapsto \Delta_{s_2}$) sse $\exists x \mid \Delta_{s_1} \leq_k \Delta_x \wedge \Delta_x \mapsto \Delta_{s_2}$.

A fase de crise gera uma intervenção do oráculo, oferecendo dados que forçam o aprendiz a questionar seu próprio conhecimento e reconhecer a necessidade de uma nova aprendizagem, usando os antigos dados acrescidos de outros novos.

A fase de revolução resulta em uma profunda reformulação das heurísticas e da linguagem de representação, levando em consideração as críticas realizadas pelo mestre durante a evolução

normal.

Esta abordagem se justifica, por um lado, pela incompletude da linguagem de descrição, e, por outro lado, pelos aspectos evolutivos e hierárquicos do conhecimento.

Neste ciclo, a crise pode resultar das seguintes percepções:

- uma inadequação da heurística utilizada na produção do conhecimento; ou
- a necessidade de se enriquecer a linguagem de representação introduzindo novos conceitos, cujos significados devem ser fornecidos ou descobertos.

3.5 Mecanismos de Raciocínio Segundo a TSE

As seções seguintes descrevem os mecanismos encarregados pela geração de novos conhecimentos, segundo a TSE.

3.5.1 Abdução

Na TSE, a abdução visa o enriquecimento de um modelo pela proposição de novas regularidades (fatos). O método de abdução deve ser capaz de extrair (ou descobrir) regularidades presentes na amostra.

No SAID, o Aprendiz utiliza a abdução para gerar um conjunto de sentenças de assimilação e um conjunto de sentenças de discriminação. Esses conjuntos são construídos utilizando-se certas heurísticas para a determinação da forma que as sentenças tomarão (por exemplo, a forma de um caminho no grafo, profundidade mínima e máxima, etc.).

O sistema S30 que será apresentado no próximo capítulo, implementa um método de busca de similaridades em objetos estruturados, sendo portanto o responsável pelo mecanismo de abdução. Deixaremos então para o próximo capítulo pormenorizações sobre os passos realizados por este mecanismo.

3.5.2 Indução

A Indução age enriquecendo um modelo pela proposição de novas hipóteses e de novas heurísticas. A indução é realizada em três passos:

1. Escolha dos fatos relevantes (regras).

2 Agregação destas regras, construindo assim regras mais gerais (as hipóteses) .

3 Agregação de hipóteses para formar uma conjectura.

3.5.2.1 Primeiro passo: escolha dos fatos relevantes

No SAID, cabe ao Mestre escolher entre os fatos gerados pela abdução aqueles por ele considerados relevantes para a construção do conceito a ser aprendido. Esta operação é um tipo de indução, visto o seu caráter generalizante.

3.5.2.2 Segundo passo: geração de hipóteses

Diversos sistemas de aprendizagem foram definidos para a aprendizagem de conceitos a partir de um conjunto de objetos [COHEN 1982; CARBONELL 1989]. O que os distingue é a forma de como representar e construir o conhecimento.

E. Mephu Nguifo [NGUIFO 1994] concebeu um sistema de aprendizagem (LEGAL) que, a partir de um conjunto de objetos (exemplos e contra-exemplos) de um conceito, gera outro conjunto de objetos (a partir das similaridades encontradas no primeiro conjunto) usando um método de generalização. Para isso, utiliza a noção de reticulado de Galois [BORDAT 1986; WILLE 1982,1991,1992]. A maior vantagem de LEGAL é sua exaustividade, dando ao sistema o espaço máximo de regularidades para ser explorado.

LEGAL utiliza a lógica proposicional para descrever os objetos e suas similaridades. Essa descrição é originalmente dada através de um conjunto finito de atributos binários caracterizando o conceito a ser aprendido, inicialmente representado por uma tripla $<O,A,I>$, onde O é um conjunto de objetos, A é um conjunto de atributos e I é uma relação binária de O para A.

Um objeto é descrito como um conjunto finito de atributos. LEGAL constrói um semi-reticulado contendo similaridades válidas, descritas pelos atributos que caracterizam A. Uma propriedade fundamental de LEGAL é que todos os objetos são descritos pelo mesmo conjunto de atributos.

O segundo passo da indução, porém, não caracteriza objetos por seus atributos, mas sim por sentenças dominantemente válidas (fatos gerados pelo mecanismo de abdução): a amostra.

A escolha das heurísticas que definirão a navegação no espaço de hipóteses deve ter uma forma de avaliar a qualidade de cada hipótese. Por exemplo, duas medidas de qualidade podem ser consideradas:

- **Simplicidade**: O número de fatos que compõem uma hipótese não deve ser grande, o que dificultaria o julgamento de sua relevância. Como LEGAL não dá suporte a esse tipo de restrição, algumas heurísticas podem ser utilizadas para impor um limite ao número de fatos que irão compor a hipótese. Assim, uma hipótese complexa gerada por LEGAL poderá derivar um conjunto de hipóteses, cada uma tendo agora sua relevância mais facilmente avaliável.
- **Cobertura**: Ainda para assegurar que somente hipóteses relevantes sejam geradas, o número de componentes de objetos que são cobertos por cada hipótese deve ser mínimo.

Este passo é aplicado tanto para os fatos de assimilação quanto para os fatos de discriminação, gerando assim dois conjuntos distintos de hipóteses. As hipóteses geradas devem obviamente serem elas próprias dominantemente válidas.

3.5.2.3 Terceiro passo: geração de conjecturas

A partir dos conjuntos de hipóteses de assimilação e de discriminação, este passo gera conjecturas sobre o conceito a ser aprendido.

Pelos mesmos motivos já expostos no segundo passo da indução (geração de hipóteses), pode-se continuar a se obedecer aos critérios de simplicidade e cobertura. Aqui, porém, a cobertura deve ser máxima, visto que as conjecturas devem identificar o maior número possível de objetos da amostra (fazendo uso do maior número de fatos conhecidos sobre os objetos), e evitando ao máximo o silêncio do aprendiz sobre novos objetos a serem classificados.

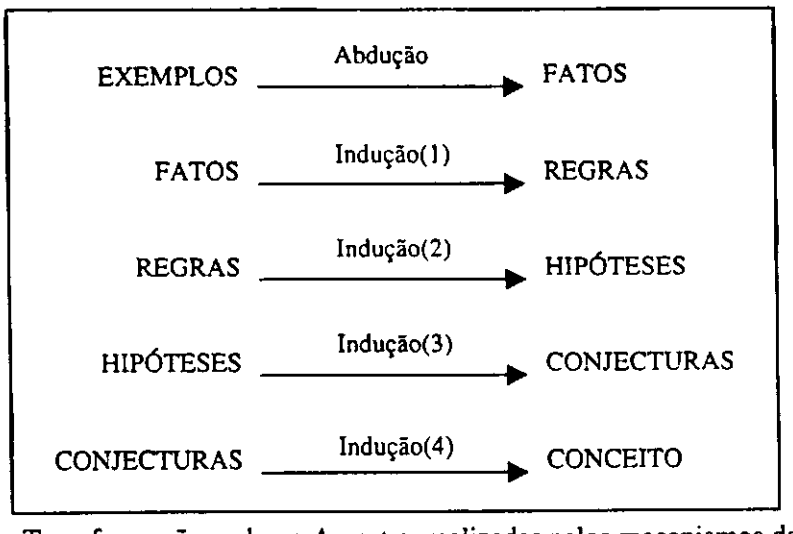


Fig. 3.8 - Transformações sobre a Amostra, realizadas pelos mecanismos da TSE

Podemos notar que o mecanismo de indução responsável pela escolha dos fatos relevantes, geração de hipóteses e geração de conjecturas é basicamente o mesmo. Neste caso, um só procedi-

mento de indução poderá processar essas três fases de indução.

Com isso, a transformação do conjunto inicial de exemplos e contra-exemplos de objetos do conceito a ser aprendido, em um conceito aprendido, terá as transformações mostrados na Fig. 3.8. Como já dissemos, este trabalho respaldará o mecanismo de abdução, que é responsável pela primeira destas transformações. O mecanismo de indução, que é o responsável pelas demais transformações, embora seja único, será mostrado neste trabalho de forma bastante superficial. Para nossos propósitos, esta fase da aprendizagem será reduzida, fazendo uma espécie de ponte entre as etapas FATOS - CONCEITO.

3.6 Considerações sobre Formulação de Hipóteses

Vamos agora iniciar uma pequena discussão sobre um dos objetivos básicos do SAID, a formulação de hipóteses; esta discussão vem a complementar o entendimento sobre a informação manipulada num sistema de aprendizagem.

A formulação de hipóteses é a atividade básica da pesquisa científica; Einstein admitia esta atividade como uma característica própria da mecânica do conhecimento humano [MORAIS 1988]. Existem atualmente duas principais escolas divididas pelas ideias com respeito à sua natureza:

- **Humana**: alguns estudiosos creditam a características inerentes ao ser humano, como a espontaneidade e a capacidade criativa, a capacidade de formular Hipóteses.
- **Artificial**: outros afirmam que se trata de uma tarefa puramente artificial, que segue um desenvolvimento lógico.

Podemos conceituar hipótese como sendo uma proposição ou conjunto destas, assumidas sem comprovação experimental. A validade da hipótese não é uma condição para a existência da mesma, mas sua formulação lógica, sim. Um enunciado hipotético desprovido de argumentação lógica é quase sempre inútil. Seguindo este raciocínio, podemos enumerar algumas qualidades pertinentes a uma boa formulação de um enunciado hipotético. Essas qualidades não são exigências básicas, porquanto não existem normas rígidas para tal construção, mas são orientações do senso comum comprovadamente presentes em boas formulações de enunciados hipotéticos [MORAIS 1988].

- **Simplicidade.** Segue o princípio cartesiano "Pensando com clareza, diz-se com simplicidade". A expressão do enunciado deve ser o mais claro possível, sem perda de conteúdo, evidentemente. A simplificação de qualquer processo exige completude, transparência, exatidão e inteligência. Podemos medir o grau de inteligência de uma pessoa ou processo

pelo grau de simplicidade com que este se expressa.

- **Adequação da linguagem.** Existem várias linguagens para a representação dos enunciados hipotéticos. Uma linguagem com a qual se deseja representar um determinado enunciado deve ser suficiente para tal. Alguns termos pertinentes a um determinado fato possuem um alto nível de abstração, o que exige a utilização de uma linguagem que possua alto poder de expressão sintática e semântica.

- **Condição de realidade:** Como vimos acima, a validade da hipótese não é uma condição para a sua existência, mas sua formulação lógica, sim. Qualquer enunciado hipotético desprovido de qualquer argumentação lógica não é interessante para formular um enunciado. Raros são os casos em que suposições intuitivas não experimentadas contribuíram para um enunciado hipotético provado válido. A expressão da realidade deve sempre ser verificada sob o ponto de vista da hipótese. Observe-se que existirão casos onde uma proposição inicialmente tomada como provisoriamente válida, foi verificada inválida após certas experimentações. Esses resultados também são importantes, pois o desmentido de uma hipótese também é um resultado a ser considerado. Esta condição não se aplica a todas as situações, podendo ser descartado quando o fenômeno a ser representado possuir restrições de representação importantes.

- **Propriedade de delimitação de campo:** A hipótese deve ser o elemento delimitador da área da atividade científica. O conteúdo de uma proposição hipotética pode permear várias disciplinas. As hipóteses podem ser generalizadoras ou particularizadoras. No primeiro caso, observa-se o uso dos quantificadores "todos" ou "nenhum". No segundo caso, os quantificadores são "alguns" , "poucos", etc... O uso destes quantificadores delimitará o escopo da proposição hipotética.

- **Sintetismo:** Sempre que houver possibilidade, o enunciado hipotético deve ser curto. Esta exigência não encobre a exigência da completude.

- **Especificidade:** Em alguns casos, uma proposição hipotética pode ser dividida em um conjunto de duas ou mais proposições. A avaliação da hipótese inicial possui o mesmo valor da avaliação do conjunto gerado pela divisão. Essa divisão deve também seguir essas regras de formação. Cada uma das proposições do conjunto é uma especificação que deve ser mais facilmente avaliada isoladamente.

- **Generalidade:** É interessante que o escopo da hipótese seja sempre o mais amplo possível. Isto nem sempre é possível, principalmente quando as proposições fogem das áreas mais próximas da Matemática para áreas mais subjetivas. As hipóteses generalizantes válidas são geralmente mais facilmente verificáveis que as hipóteses subjetivas.

3.7 Resumo do capítulo

Apresentamos neste capítulo uma visão geral do Agente Racional SAID, abordando seus aspectos estruturais e funcionais. Apresentamos o princípio da Lógica das provas e refutações e sua utilidade para a descoberta científica e resolução de problemas. Apresentamos ainda o protocolo MOSCA, onde estes tópicos são associados ao processo de identificação de conceitos. Apresentamos ainda os mecanismos de inferência deste protocolo (abdução, indução e dedução) no contexto da Teoria Semi-Empírica, que nos permitiu caracterizar o papel do mecanismo de abdução, sua relação com a descoberta e com os demais mecanismos (indução e dedução). Mostramos ainda que o sistema SAID pode ser visto como um sistema popperiano, apto a realizar descobertas.

Vimos que o mecanismo de abdução pode ser implementado a partir da identificação de similaridades verificadas nos objetos de uma amostra nos sistemas de aprendizagem por exemplos. Por fim apresentamos, numa breve discussão, o que pode ser tomado como regras gerais para formulações de hipóteses, que podem ser efetivamente obedecidas pelos usuários do SAID.

Com este capítulo, encerramos as discussões dos principais conceitos que servem de base ao sistema S30 apresentado no próximo capítulo.

Capítulo 4

O Sistema S3O

4.1 Introdução

Esse capítulo apresenta o sistema S3O (*Similarity Search on Structured Objects*). Trata-se de um sistema de aprendizagem conceitual que toma como base o método INNE [LIQUIÈRE 1990], apresentado no Capítulo 2. Esse novo sistema identifica similaridades[1] entre os componentes elementares de objetos definidos estruturalmente e analisa o montante de ocorrências similares, a fim de detectar regularidades[2] que permitam construir um conjunto de hipóteses que formarão conjecturas sobre um conceito (objeto generalizante) desses objetos.

Será inicialmente apresentada a posição do S3O no contexto da Aprendizagem de Máquina, a partir dos conceitos expostos nos capítulos anteriores. Logo após, será apresentado o método sob uma representação baseada em grafos, especificando funcionalmente cada fase do mesmo. Veremos que dentre os algoritmos que fazem parte desse sistema, existem dois algoritmos clássicos para os problemas de *busca em profundidade* e *casamento de padrões*, que foram adaptados às nossas necessidades com o objetivo de alcançar um bom desempenho.

4.2 Caracterização do S3O como sistema de aprendizagem

Apresentamos nos capítulos 1 e 2 definições sobre vários aspectos e parâmetros dos programas de aprendizagem de máquina; de acordo com o que foi então apresentado, podemos classificar o S3O como segue:

- **Linha de pesquisa**: O S3O realiza a aprendizagem conceitual, que o caracteriza como uma ferramenta de *análise teórica* sobre conjuntos de dados.
- **Principal paradigma da aprendizagem de máquina utilizado**: O paradigma da aprendizagem inerente ao S3O é a *aprendizagem a partir exemplos.*
- **Representação do conhecimento**: No S3O, o conhecimento é representado pelas con-

[1] *Similaridade* é aqui considerada como uma instância em que os objetos envolvidos apresentam a condição de igualdade em aspecto.

[2] *Regularidades* são aqui consideradas as similaridades que, com certa freqüência, ocorrem entre objetos.

venções utilizadas nas *teoria semi-empíricas* (Cap. 3).

- **Linguagem de representação do conhecimento**: O conhecimento é expresso por uma linguagem gráfica, próxima aos *Grafos Conceituais.*
- **Processo de definição de classes**: O S3O define classes estruturalmente.
- **Classificação segundo os requerimentos gerais dos métodos de aprendizagem**:
  - **Generalidade.** A generalidade do método é verificada pela sua característica de *aplicabilidade a vários domínios*, onde a restrição imposta ao contexto é definida apenas pela característica de representação dos seus objetos, que devem ser estruturados.
  - **Intensionalidade.** Esta característica é definida pelo objetivo-fim do método - A *aprendizagem de conceitos*; em qualquer caso, como visto anteriormente, a resolução de problemas pode ser transformada em formulações de conceitos.
  - **Reatividade.** Independentemente do contexto do domínio, o produto do sistema (as hipóteses geradas) é uma conseqüência direta (função) do domínio; ou seja, o *método reage ao domínio.*

4.3 O SAID e o S3O

4.3.1 A Interseção SAID/S3O

Como vimos no capítulo anterior, o agente racional SAID emprega o sistema S3O no seu agente Aprendiz, que o utiliza como uma de suas ferramentas de aprendizagem. O S3O suporta o mecanismo básico de abdução.

O mecanismo de abdução é, como mostrado no capítulo anterior, o responsável pela transformação dos exemplos em conjuntos de fatos que formarão hipóteses sobre um conceito, através da busca de similaridades, utilizando a lógica majoritária. Podemos observar esta afirmação de maneira bastante superficial, associando a definição lógica desse mecanismo ao caso da aprendizagem por exemplos.

Premissa básica: Todos os objetos de um determinado conceito C possuem um conjunto S de determinadas similaridades,

Fato verificado: Os objetos de um conjunto H possuem as similaridades contidas em S.

Hipótese obtida: Os objetos do conjunto H pertencem ao conceito C

Desta maneira, a função do S3O é construir o conjunto *H*. Observe-se que quanto maior esse conjunto (*H* tem cardinalidade máximo = cardinalidade do conjunto *C*), mais consistente será a hipótese obtida.

O mecanismo de indução, que é o responsável pelas demais transformações e será mostrado neste trabalho de forma superficial. Para nossos propósitos, esta fase da aprendizagem será reduzida, e a construção do objeto generalizante será apresentada de maneira ilustrativa.

4.3.2 Requisitos para o S3O

Vamos inicialmente enumerar as exigências do SAID para o sistema de aprendizagem S3O, segundo os principais tópicos discutidos nos capítulos 2 e 3. Podemos enumerar os seguintes requerimentos para o S3O como sistema de aprendizagem:

- **Estratégia de Aprendizagem**: a estratégia utilizada deve ser a *Busca de similaridades* nos fatos que formam os objetos da amostra (domínio).
- **Quantidade de Inferências ou Nível Inferencial**: o nível inferencial empregado pode ser considerado *alto*, visto que ao implementar um mecanismo abdutivo, o sistema deve ser capaz de realizar descobertas.
- **Tipo do conhecimento adquirido**: o conhecimento adquirido pelo sistema deve ser formado pelas *hipóteses* que formarão *conjecturas* sobre um determinado *conceito*. Estas hipóteses devem ser geradas em dois diferentes conjuntos, de acordo com a interpretação sobre o conjunto de fatos; uma que permita a interpretação das hipóteses sobre toda a amostra (assimilação) e a outra que inclua restrições aos exemplos portadores de ruído ou exceções (discriminação).
- **Domínio da aplicação**: o sistema deve prover um método genérico de aprendizagem (*aprendizagem cognitiva*)
- **Tipo de domínio:** o sistema pode aprender sob *domínios estáticos* ou *domínios dinâmicos*, pois o conjunto de treinamento pode variar, mas a apresentação da amostra captura um "momento estático" do domínio .
- **Tipo de exemplos apresentados**: os exemplos apresentados ao sistema devem ser *instâncias positivas* (exemplares perfeitos) e *instâncias negativas* (objetos portadores de ruído ou que não pertencem ao conceito) de exemplos do conceito a ser aprendido.
- **Tipo de Ruído presente nos exemplos**: o ruído apresenta-se na forma de *distorções estruturais* que podem ser encontradas sempre nas descrições dos objetos da amostra, inde-

pendentemente da sua origem (deformidade natural do exemplo, má representação do exemplo, etc).

- **Fonte dos exemplos**: Os exemplos são fornecidos pelo usuário.
- **Lógica utilizada para validação dos fatos**: o sistema deve utilizar a *lógica majoritária* para validar os fatos verificados.
- **Estratégia de exploração do espaço de generalização**: a estratégia de exploração deve utilizar *critérios de seleção* definidos pelo aprendiz. Esses critérios devem incluir um limite positivo de validade majoritária (número mínimo de fatos que deverão ser verdadeiros para validar sua agregação), um limite negativo de validade majoritária (número mínimo de fatos que devem ser falsos para falsificar a agregação dos mesmos), um filtro de atributos para os exemplos do conceito e um limite máximo para o comprimento dos fatos a considerar.
- **Algoritmo de apresentação de exemplos**: o sistema deve descartar sua base de conhecimento; ele deve reconstruí-la a cada vez. O algoritmo é então, *não-incremental*.

4.4 Formalismo para o S3O

4.4.1 Objetos e Classes

Vamos agora apresentar as definições dos elementos descritivos básicos, a saber, conceitos de objeto e classe.

- **Objeto** é um conjunto de propriedades ou fatos ao qual está associado um identificador, sendo esse conjunto de fatos suficientes para caracterizá-lo. O conhecimento sobre um objeto é obtido pela expressão de seus fatos constituintes numa linguagem de descrição adequada. O valor dos fatos podem variar de objeto para objeto. Um objeto pode ser constituído a partir de outros objetos; um objeto O_2, com base de fatos F_2 é constituinte de um objeto O_1 com base de fatos F_1, quando $O_2 \subseteq O_1$.
- **Classe** é uma estrutura que permite o agrupamento de objetos com fatos semelhantes. Um objeto O_1 pertence à mesma classe que outro objeto O_2 quando $O_1 = O_2$. Um objeto O_1 é idêntico a um outro objeto O_2 quando $V(O_1) = V(O_2)$, sendo $V(O_i)$ o conjunto de valores dos fatos que compõem o objeto i.

4.4.2 Discriminação e Assimilação:

Consideremos dois tipos de interpretação sobre os dados:

- **Assimilação.** Esse tipo de interpretação é identificado quando uma questão (função de busca) sobre um determinado conjunto de elementos abrange em sua resposta, todos os elementos do conjunto. Uma assimilação parcial é o resultado de uma busca que obtém como resposta quase todos os objetos do conjunto.
- **Discriminação.** Considere dois conjuntos de objetos E e C; Esse tipo de interpretação será identificado quando o resultado de uma questão fornecer um resultado R, tal que $R \subseteq E$ e $R \cap C = \emptyset$.

  A discriminação parcial é observada quando o resultado da questão abrange quase todos os objetos do conjunto E e raros objetos do conjunto C.

4.4.3 Intensão e Extensão

Consideremos dois tipos de classes:

- A **Classe Intensional**[3] é definida em intensão sobre um conjunto de fatos e um mecanismo de decisão. Esta classe é formada pelos objetos validados pela função de decisão.
- A **Classe Extensional**[4] é definida em extensão. E esta classe é definida pelo conjunto de instâncias associadas à mesma. Esse conceito assemelha-se ao conceito do mesmo nome no campo da física, onde significa as propriedades da matéria pela qual os corpos ocupam uma parte do espaço.

4.4.4 Objetivo do S3O

O objetivo do S3O é a construção em intensão, de uma classe definida em extensão. Ou seja, o conjunto Amostra é composto de elementos que são Instâncias positivas ou negativas da Classe Principal; estas instâncias estão descritas em termos de conjuntos de valores dos atributos característicos dos objetos da classe que representam, descritos extensionalmente . O S3O deve gerar uma descrição para esses objetos sem expressar valores tangíveis dos atributos dos objetos, mas generalizá-los em sua intensão, expressando os mesmos em termos de suas classes e suas relações.

---

[3] A intensão expressa as propriedades conotadas por um conceito.

[4] A extensão é a classe de objetos denotadas por um conceito específico.

4.5 Descrição estrutural

Vamos, inicialmente, definir para o S3O, um formalismo para descrição. Seguiremos o formalismo semelhante ao que descreve o INNE [LIQUIÈRE 1990], com algumas modificações importantes.

4.5.1 Redução da Complexidade do Algoritmo de Aprendizagem

De maneira geral, os métodos de aprendizagem que usam descrições estruturais de objetos, enfrentam o problema da complexidade relacionada à linguagem de descrição [MICHALSKI 1983, GASCUEL 1988]. Podemos reduzir os efeitos da complexidade associada, utilizando uma estratégia de estabelecer restrições em pelo menos duas frentes:

- Nas declarações que possam ser encontradas e
- Nos tipos de objetos a serem considerados.

Empregando essa estratégia podemos, na maioria dos casos, obter uma relevante redução nas necessidades de tempo e espaço, pela redução do número de operações do programa de aprendizagem. Sabemos antecipadamente que tais restrições devem ser suficientemente gerais para prover uma linguagem de descrição poderosa, suficiente para nossos propósitos, mas também suficientemente particular, para reduzir os efeitos da NP-completude, tanto no processo da aprendizagem quanto no processo da decisão.

4.5.2 Grafos Conceituais

4.5.2.1 Definição de Grafo Conceitual

Utilizaremos no S3O uma variação de *grafos conceituais* [SOWA 1984] como descritor. Outras representações poderiam ser utilizadas, como é o caso das redes semânticas [FINDLER 1979]. Uma representação interessante poderia ter sido utilizar *linguagem objeto* [WINSTON 1979]. Escolhemos, porém, a representação gráfica, baseados na sua principal vantagem: estruturas descritas graficamente são entendidas com maior facilidades de que através de qualquer outro recurso de descrição. A descrição em grafos é, obviamente, um formalismo. Os programas de aprendizagem não podem ser escritos por meio desse recurso.

Utilizaremos também uma segunda representação dos grafos conceituais mais próxima da possível estrutura de dados a ser utilizada em implementações desse método; isto tornará mais fácil a apresentação e o entendimento dos procedimentos utilizados. Assim, utilizaremos o primeiro des-

critor para apresentar o método em seus níveis mais altos de abstração, e o segundo nos níveis mais baixos, durante a apresentação dos algoritmos. Essa representação é um importante diferencial da utilizada por Liquière. O INNE manipula separadamente os objetos e as relações; nosso método utiliza como estrutura básica uma tupla <Objeto,Atributo,Valor>, uma relação entre objetos. Veremos mais tarde que isto implicará em um aumento na eficiência do método em relação ao INNE.

Um *grafo conceitual* é uma estrutura que permite descrever um objeto a partir de seus objetos-constituintes e das relações entre os mesmos. Em outras palavras, podemos dizer que os grafos conceituais permitem estruturar os objetos em classes.

Um *grafo conceitual* é um grafo finito, conectado, rotulado e direcionado **[SOWA 1984]**. Os vértices de um grafo conceitual são particionados em *Conjunto de objetos* e *Conjunto das relações*. Sowa descreve esses conjuntos como *Conjunto de Conceitos* e *Conjunto de Relações Conceituais*, respectivamente.

Uma outra diferença sutil da variação de grafo conceitual aqui empregada é a substituição da representação do conjunto de relações. Para Sowa **[SOWA 1984]**, essas relações são representadas por um círculo, como mostra a Fig. 4.1. Preferimos utilizar um arco de ligação rotulado entre os objetos, aproximando mais ainda das redes semânticas, como mostra a Fig. 4.2.

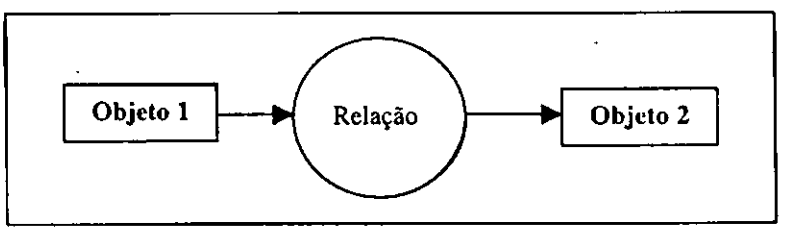


Fig. 4.1 - Estrutura básica de um Grafo conceitual **[SOWA 84]**.

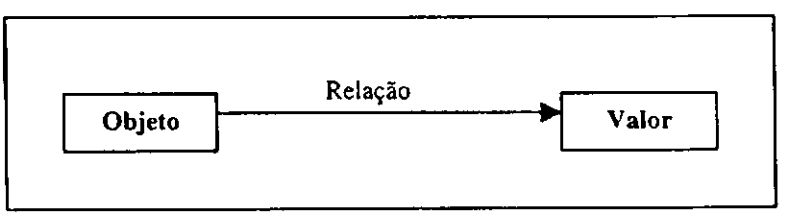


Fig. 4.2 - Variação da representação Objeto-Atributo-Valor

4.5.2.2 Vantagens da Descrição

Podemos, inicialmente, enumerar as principais vantagens desta representação:

- **Facilidade de expressão:** A expressão do conhecimento se faz através de grafos, cujo

entendimento é bastante simplificado; o objetivo do emprego dos grafos é oferecer, em termos, uma base de expressão da linguagem natural; com esta representação, permite-se a fácil compreensão do conhecimento expresso.

- **Definição do conhecimento**: A gestão dos grafos conceituais permite a introdução de conhecimentos como as taxonomias sobre os objetos e as restrições locais a uma relação existente entre objetos.

- **Declaração natural dos mecanismos básicos da Lógica (Instanciação e Implicação):** Esses mecanismos são facilmente declarados com esse tipo de representação de conhecimento, assim como as noções básicas da aprendizagem (*Especialização* e *Generalização*) que serão abordadas logo a seguir. Com isso, uma declaração é dita ser verdadeira sobre um objeto se e somente se o grafo da declaração é um sub-grafo do grafo do objeto. Consequentemente, o problema associado transforma-se em um "casamento de padrões" nos passos de aprendizagem e aplicação.

4.5.2.3 Ordem Parcial

Uma ordem parcial pode ser estabelecida nessa descrição se pudermos estabelecer e utilizar os seguintes mecanismos:

- **Especialização**: Dizemos que U é uma especialização de V com a notação $V \geq U$, se o grafo U pode ser obtido a partir de um grafo V através de certas regras de formação aplicadas de maneira que: {conjunto de atributos de V} $\subseteq$ {conjunto de atributos de U}, ou seja, U é uma subclasse de V.

- **Generalização**: Dizemos que V é uma generalização de U se o grafo U é uma especialização do grafo V. Isto é, V é uma classe superior de U.

Vamos agora estabelecer a base da descrição. Seja um conjunto E de palavras, um subconjunto T de E parcialmente ordenado, denominado conjunto dos tipos e um subconjunto R de E denominado conjunto das relações. A qualquer palavra de E corresponde um só tipo de T e uma só relação de R. Assim, $E = T \cup R$ e $T \cap R = \emptyset$. A ordem parcial sobre T permite definir uma *taxonomia de tipos.*

Para deixar mais claro esses conceitos, vamos considerar o seguinte exemplo. Suponha que queiramos utilizar nossa versão de grafo conceitual para representar nosso conhecido exemplo de ARCO (Fig. 4.3). A taxonomia associada[5] é mostrada na Fig. 4.4.

---

5 Os objetos estão aqui descritos no plano R^2 e o conceito de estabilidade é indêntico ao de R^3

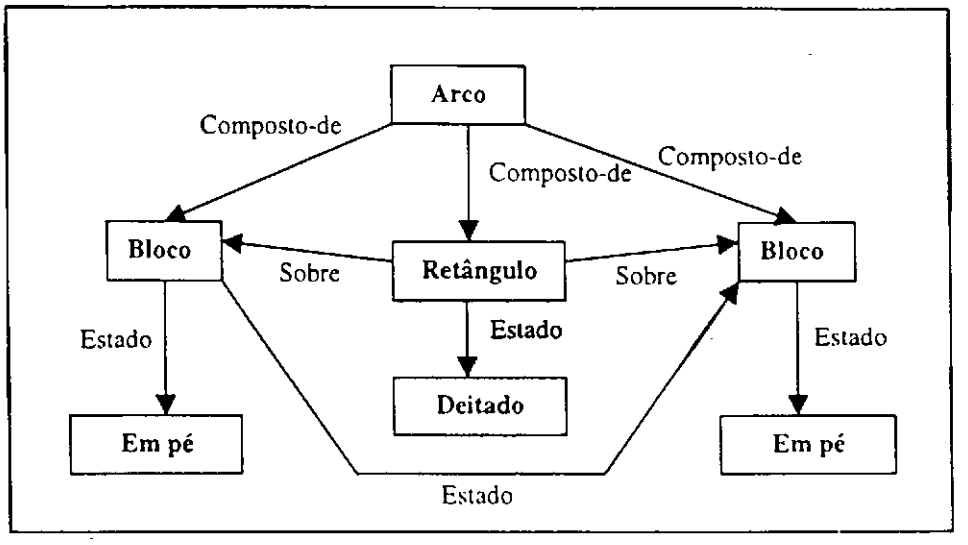


Fig. 4.3 - Representação do conceito ARCO em um grafo conceitual.

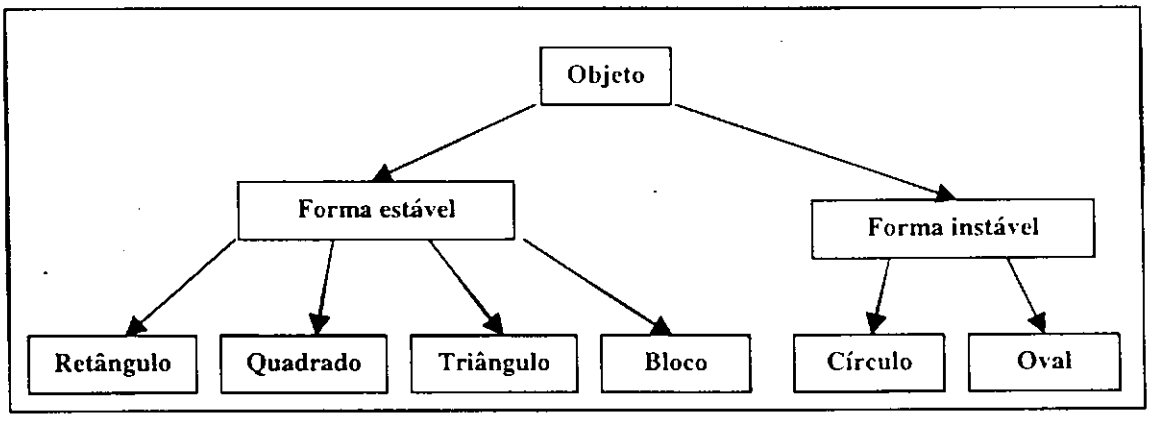


Fig. 4.4 - Uma Taxonomia do domínio do mundo dos blocos.

4.5.2.4 Estrutura do Grafo Conceitual

Observando com maior clareza essa descrição, podemos identificar as partes constituintes do grafo conceitual; seu conjunto de vértices é dividido em:

- **Conjunto de Objetos**: Vértices de objetos são rotulados pelo par (*tipo*, *referente*) e são representados pelos retângulos, onde:

  - **Tipo** é uma palavra pertencente a um conjunto T, que determina a classe vinculada ao objeto, **e**

  - O **Referente** pode ser:

Uma *constante* (objeto individual), como por exemplo, (Avião, "Cessna 175"), (Carro, "Ford Mondeo") ou (Cientista, "Marie Curie"). Esse é o caso do nosso exemplo, onde figura um conjunto de constantes com todos os tipos do conceito;

Um *conjunto de constantes* com todos os tipos do conceito, como por exemplo: (Cientista, "Marie Curie", "Albert Einstein", "Jacques Custeau");

Genérico (*), que corresponde a todos os objetos de um tipo: Por exemplo, (Países, *). * refere-se a todos os países. Ou seja, o referente genérico é uma Classe.

- **Relações** são representadas por uma seta rotulada por uma palavra, ligando dois objetos, indicando a relação.

Temos uma ordem parcial em T. esta ordem pode ser representada por uma taxonomia de tipos, assim como o exemplo mostrado na Fig. 4.4.

4.5.2.5 Árvore Conceitual, Trilha Conceitual e Vértice Conceitual

Vamos agora definir três tipos muito particulares de grafos conceituais: Uma *Árvore Conceitual* é um grafo conceitual que possui uma raiz rotulada representando um conceito e um conjunto de vértices, que representam classes, objetos ou instâncias destes ligados por conexões simples (não cíclicas), representando as relações entre os mesmos. Uma *Trilha Conceitual* é uma árvore conceitual cujos vértices possuem no máximo um predecessor e um sucessor e um vértice conceitual é o conjunto de trilhas conceituais entre um nodo da árvore e uma de suas folhas. Na Fig. 4.5 é apresentado um exemplo de trilha conceitual, representado pelos objetos com traçados enfatizados e um dos principais vértices conceituais em destaque.

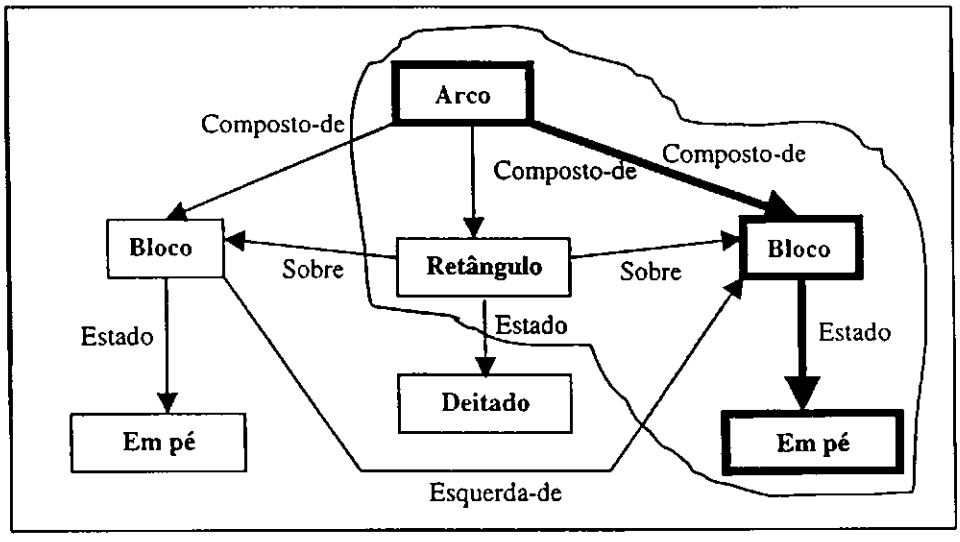


Fig. 4.5 - Trilha Conceitual e Vértice Conceitual.

4.5.2.7 Regras de Formação para Grafos Conceituais.

A formação de um grafo conceitual corresponde à construção da representação de um objeto estruturado. Não se trata de uma construção empírica, pois deve seguir uma metodologia contendo regras de formação. A construção de tal representação sempre será iniciada pela identificação dos componentes básicos de um objeto e sua representação em grafos simples. A partir daí, podemos utilizar quatro regras básicas de formação necessárias para derivar um grafo conceitual *W* a partir de dois outros grafos *U* e *V*:

1. **Cópia** de um grafo *U* em um grafo *W*, que é uma duplicação do grafo *U*.
2. **Simplificação ou Unificação:** Se duas relações *r* e *s* sobre um grafo *U* possuem um mesmo rótulo e também os mesmos predecessores e sucessores sobre o grafo *U*, então elas são consideradas redundantes e portanto devem ser unificadas.
3. **Restrição**: para todo vértice conceitual *c* de *U*, *tipo*(*c*) pode ser substituído por um subtipo. Se *c* é genérico, seu referendo pode ser substituído por um marcador individual de acordo com o tipo de *c*.
4. **Junção**: Se *c* de *U* e *d* de *V* são dois objetos idênticos, então podemos derivar um grafo *W* ligando ao objeto *d* no grafo *V* todas as relações de *c* em *U* e suprimir *c*.

4.6 Estrutura funcional do S3O

A Fig. 4.7 mostra o sistema S3O em relação às informações que ele manipula - os dados de entrada (descrições dos exemplos e contra-exemplos de objetos do conceito), o controle de busca (conjunto de restrições de busca) e os dados de saída (os fatos majoritariamente válidos de assimilação e discriminação).

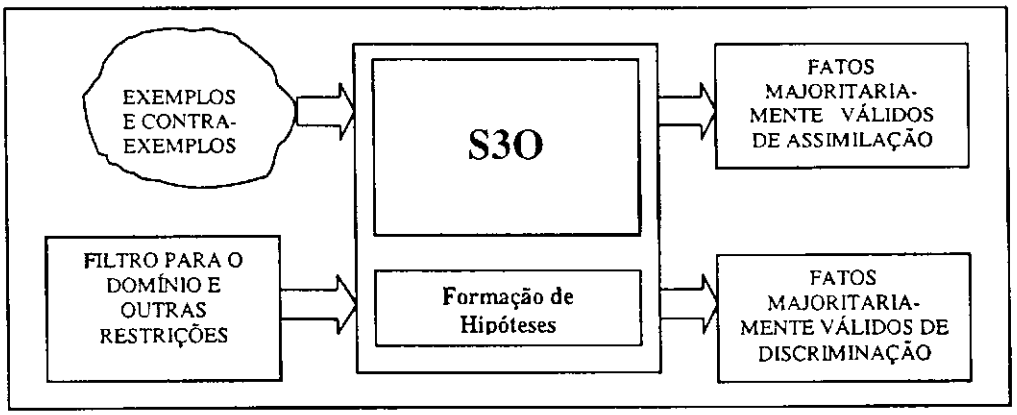


Fig. 4.7 - Estrutura Funcional do S3O

A Fig. 4.8 mostra os módulos funcionalmente distintos do S3O em ordem de ativação; essa figura inclui ainda o módulo de geração do objeto generalizante, que não faz parte do sistema S3O e é realizado por um processo indutivo que agrupa as hipóteses geradas. Passamos agora a descrever as operações realizadas por cada um desses módulos. De acordo com a estrutura mostrada na mesma figura, podemos identificar os seguintes módulos componentes da busca de similaridades:

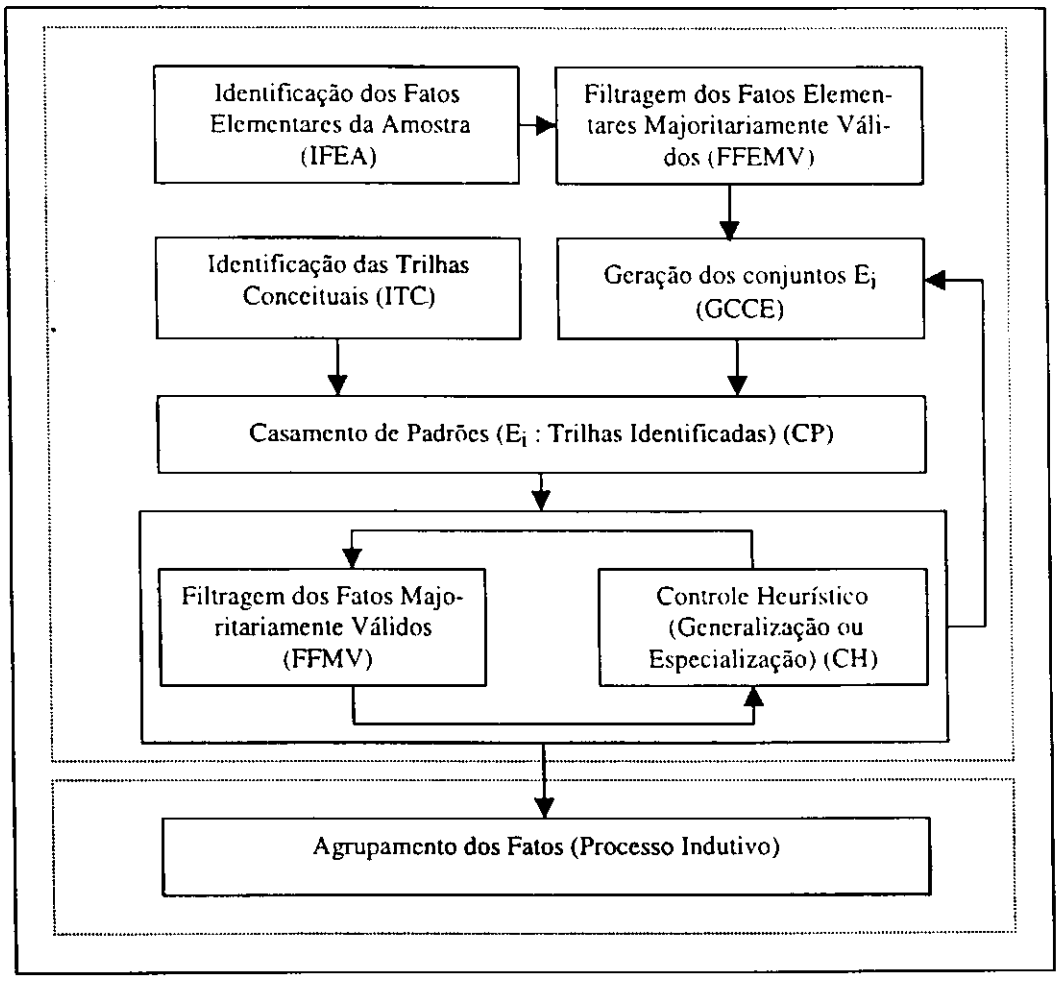


Fig. 4.8 - Módulos Funcionais do S3O

1. Identificação dos Fatos Elementares da Amostra (IFEA): Nesse módulo, o conjunto Amostra é percorrido com o objetivo de identificar suas estruturas componentes básicas, indicando as correspondências entre os fatos definidos em intensão e extensão.

2. Filtragem dos Fatos Elementares Majoritariamente Válidos (FFEMV): Tomando como entrada o resultado do módulo anterior, esse módulo filtra os Fatos Elementares identificados que ocorrem segundo os critérios majoritários impostos pelas restrições de validade majoritária, sempre mantendo a correspondência dos conjuntos definidos em intensão e extensão.

3. Identificação das Trilhas Conceituais (ITC): Esse módulo identifica para cada elemento descrito na Amostra todas as suas trilhas conceituais, identificando a existência de trilhas repetidas (redundantes).

4. Geração dos Conjuntos de Classes de Equivalência (GCCE): Esse módulo toma cada um dos elementos do conjunto gerado pelo módulo anterior, cujo tamanho elementar é igual a 1 (dois vértices conceituais ligados por uma relação), denominados E_1 , e os combina através das possíveis ligações entre si, gerando os conjuntos E_2, E_3 e assim sucessivamente. A cada conjunto gerado, o módulo de casamento de padrões é acionado para checar a existência majoritária de cada elemento de E_i. A construção dos conjuntos E_i termina quando não existe mais ocorrências dos elementos dos mesmos nas trilhas.

5. Casamento de Padrões (CP): Esse módulo toma cada um dos elementos E_i do conjunto formado pelo módulo anterior e procura sua ocorrência no corpo das trilhas conceituais encontradas, indicando cada ocorrência.

6. O módulo de Filtragem dos Fatos Majoritariamente Válidos (FFMV) funciona de maneira semelhante à filtragem dos FEMV.

7. O módulo de Controle Heurístico (CH) utiliza as funções de generalização (aumento do nível de abstração do fato) ou especificação (diminuição do nível de abstração do fato) sobre os fatos que dispõe. A aplicação das heurísticas permitem alterar o resultado da filtragem dos FMV e é realizado interativamente com o usuário. O resultado é o conjunto de hipóteses definidas em intensão e extensão que serão considerados para a construção do conceito.

8. Agrupamento dos fatos (processo indutivo): Neste módulo, os fatos majoritariamente válidos sobre a amostra são agrupados, gerando uma descrição generalizante.

Para exemplificar o funcionamento dos mesmos, vamos apresentá-los segundo a ordem de ativação já indicada, observando a cada instante, os resultados intermediários. Para isso, vamos considerar mais uma vez o exemplo do conceito de Arco.

Vamos agora considerar o conjunto Amostra do exemplo ARCO que é apresentado na Fig. 2.9 (pg. 57). A descrição das classes dos objetos da Amostra estão na Tabela 4.1 e a descrição das Instâncias para os objetos da Amostra está na Tabela 4.2.

Classe	Atributos da Classe
Arco	Tipo-de [Objeto] Composto-de [n, Formas]
Forma	Tipo-de [Objeto] Status [1,{em-pé, deitado}] Sobre [n, Formas]
Estável	Tipo-de [Forma]
Instável	Tipo-de [Forma]
Quadrado	Tipo-de [Estável]
Retângulo	Tipo-de [Estável]
Triângulo	Tipo-de [Estável]
Oval	Tipo-de [Instável]
Bloco	Tipo-de [Estável] Esquerda-de [Bloco] Sobre ["Chão"]

Tabela 4.1 - Definição de Classes para os objetos da Amostra.

Instâncias dos objetos do exemplo (Conjunto Amostra)			
Objeto	Atributos	Objeto	Atributos
Exemplo 1:	É-um [Arco] Composto-de [Quadrado1,Bloco1,Bloco2]	Bloco1:	É-um [Bloco] Status [em-pé] Esquerda-de [Bloco2]
Exemplo 2:	É-um [Arco] Composto-de [Retângulo,Bloco3,Bloco4]	Bloco2:	É-um [Bloco] Status [em-pé]
Exemplo 3 :	É-um [Arco] Composto-de [Triângulo1 ,Bloco5,Bloco6]	Bloco3:	É-um [Bloco] Status [em-pé] Esquerda-de [Bloco4]
Contra_Exemplo 1:	É-um [Arco] Composto-de [Oval1,Bloco7]	Bloco4:	É-um [Bloco] Status [em-pé]
Quadrado 1:	É-um [Quadrado] Status [deitado] Sobre [Bloco1, Bloco2]	Bloco5:	É-um [Bloco] Status [em-pé] Esquerda-de [Bloco6]
Retângulo 1:	É-um [Retângulo] Status [deitado] Sobre [Bloco3, Bloco4]	Bloco6:	É-um [Bloco] Status [em-pé]
Triângulo 1:	É-um [Triângulo] Status [deitado] Sobre [Bloco5, Bloco6]	Bloco7:	É-um [Bloco] Status [em-pé]
Oval 1:	É-um [Oval] Status [deitado] Sobre [Bloco7]		

Tabela 4.2 - Conjunto *Amostra* de Instâncias de objetos para o exemplo

Passamos agora a apresentar detalhadamente, os procedimentos do método de aprendizagem proposto.

4.6.1 A Identificação dos Enunciados Elementares

Vamos inicialmente construir o conjunto dos Fatos (ou Enunciados) Elementares da amostra. Para isso, o S3O manipula os seguintes dados:

1. Uma amostra formada de exemplos e contra-exemplos do conceito a ser aprendido;
2. Uma relação das classes, atributos e valores de interesse para a aprendizagem do conceito em questão;
3. Identificador, entre essas classes, daquele que representa o objeto composto;
4. Limites positivo e negativo de validade majoritária; e
5. Identificação do tipo de enunciados que serão gerados (assimilação ou discriminação).

Esta primeira fase deve gerar o conjunto de enunciados elementares presentes na amostra. O conjunto de objetos para o exemplo está definido como mostrado na Tab. 4.2. A classe Objeto não está definida, pois se trata da Classe Principal e é o conceito que se procura construir. A classe Forma também referencia um objeto. Nesse caso, não se trata de um conceito a construir, mas de uma função de ligação a uma classe de alto nível de abstração para a expressão de existência. Outras considerações devem ser observadas:

- Uma definição de Classe/Objeto é uma lista de atributos, iniciada pelo atributo *Tipo*, conforme mostrado na seção 4.4.2.4.
- O índice n num atributo com a estrutura: Atributo [n,{Lista de Atributos}], indica a aridade do Atributo, isto é, quantos dos valores pertencentes a {Lista de Atributos} ele pode assumir. Para simplificação da representação, um fato que possui aridade *n* superior à default (1) pode ser fatorado em *n* fatos que possuem os mesmos objetos e atributos, diferenciados pelos *n* valores diferentes.
- A característica de Herança permite que as subclasses redefinam ou omitam seus atributos, como fizemos, por exemplo, com a definição da classe Bloco, que herda as características de Forma (indiretamente), redefinindo o atributo Sobre[n, Formas] e omitindo o atributo Status[1,{em-pé, deitado}].

Uma vez definidas as Classes, resta-nos instanciar às mesmas, os objetos do conjunto Amostra. Aqui, cada elemento componente de cada objeto é identificado; para isso, usaremos nesse exemplo, um nome genérico de acordo com a classe (ex: Bloco1, Retângulo 1, ...).

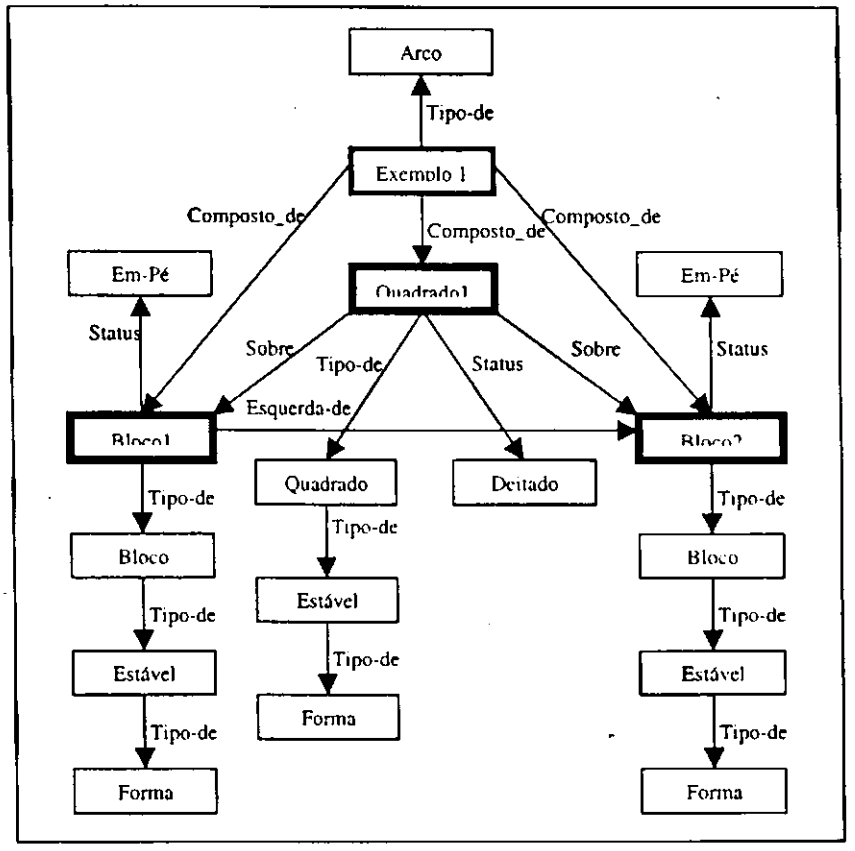


Fig. 4.9(a) - Rede conceitual para *Exemplo1*

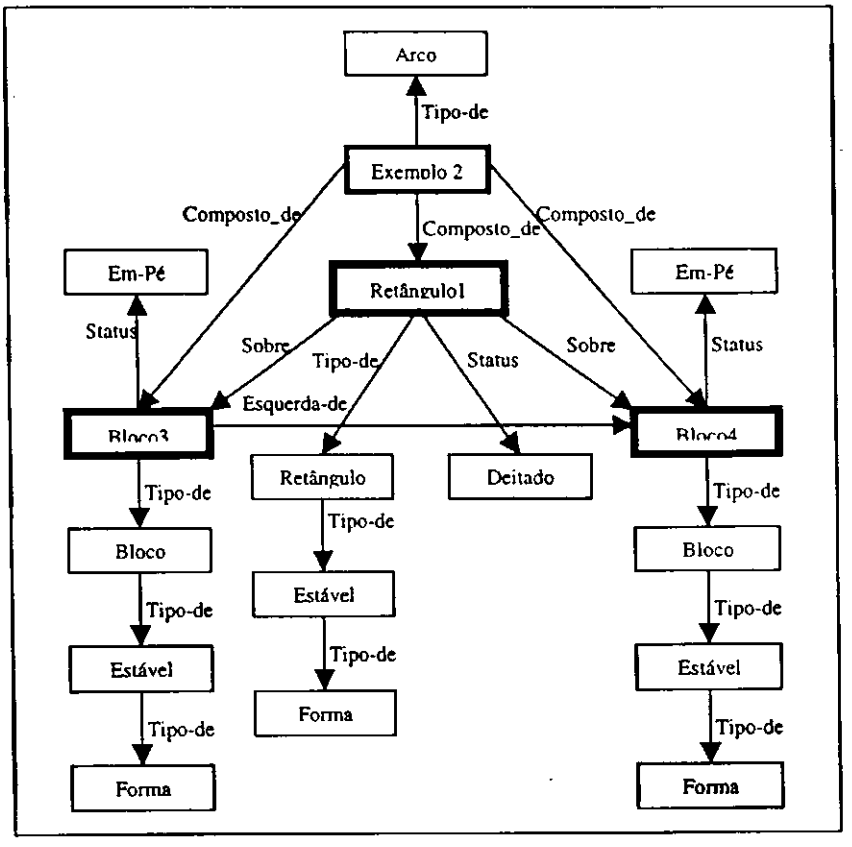


Fig. 4.9(b) - Rede conceitual para *Exemplo2*

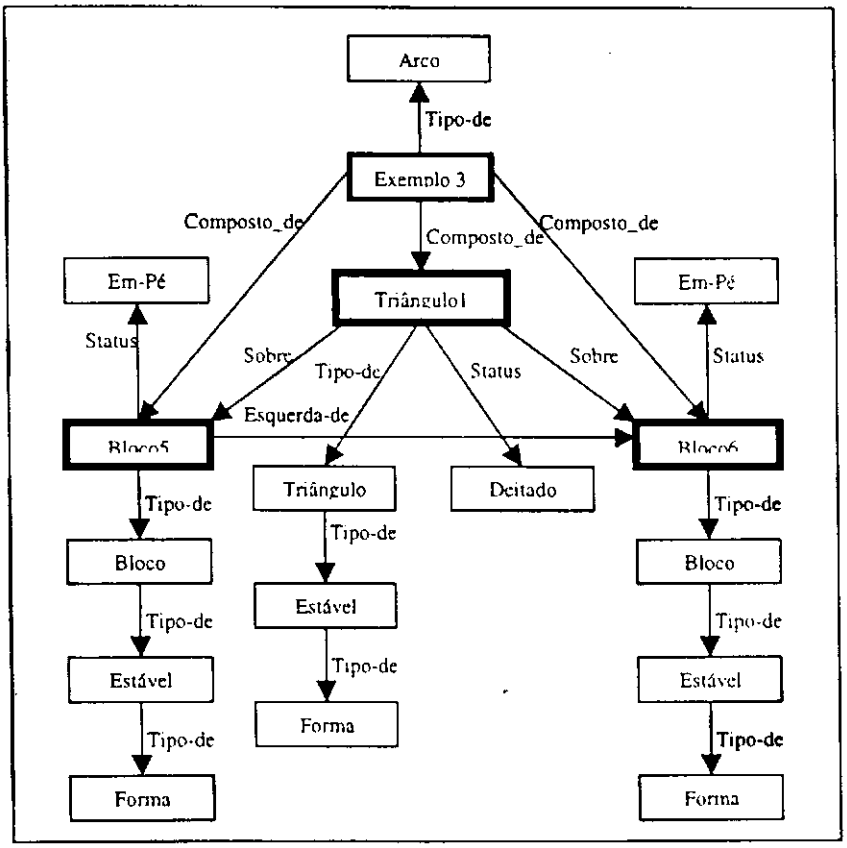


Fig. 4.9(c) - Rede conceitual para *Exemplo3*

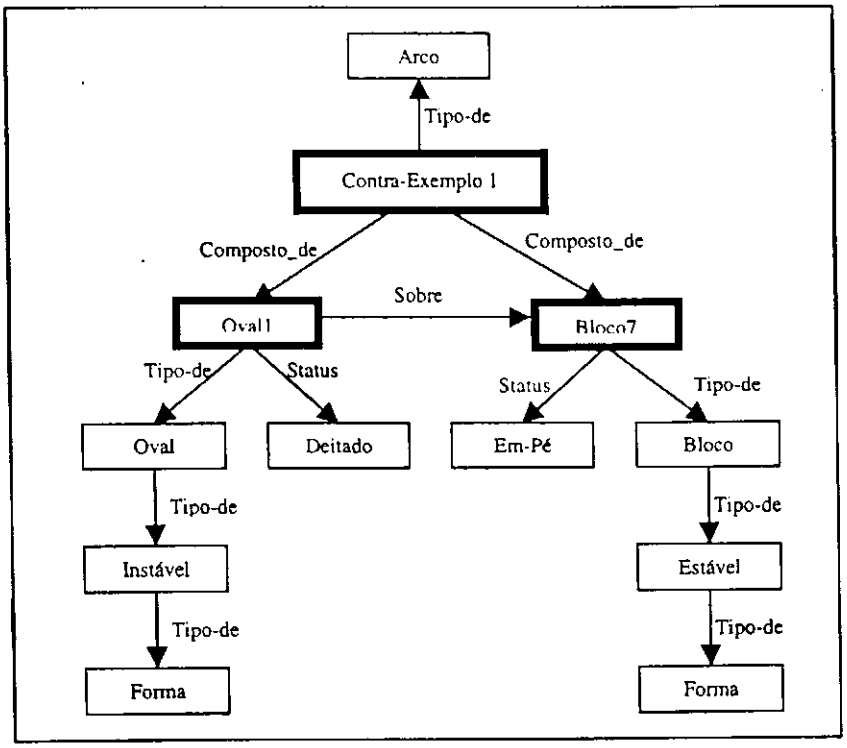


Fig. 4.9(d) - Rede conceitual para *Contra-Exemplo1*

Neste ponto, já temos todos os objetos da amostra definidos. A Amostra contém instâncias positivas (exemplos) e negativas (contra-exemplos) de objetos cujo conceito deve ser aprendido a partir das hipóteses geradas pelo S3O.

Os dados definidos até esse momento já permitem formar para cada exemplo apresentado uma rede conceitual formada pelos grafos conceituais que descrevem as classes e aqueles que descrevem as Instâncias dos objetos constituintes de cada exemplo ligados por suas relações. As Figs. 4.9(a,b,c,d) apresentam os aspectos das redes para os objetos componentes do conjunto Amostra. Podemos observar que os objetos especificados como Instâncias listados na Tabela 4.3 estão também ligados às classes respectivas de cada Atributo definidas na Tabela 4.2. Os objetos definidos como Instâncias estão destacados com contornos em negrito.

A rede conceitual apresentada na Fig. 4.9(a), apresenta os quatro objetos descritos como instâncias, algumas classes do domínio do mundo dos blocos e os atributos que definem as relações entre estas e as instâncias. Podemos notar que *Exemplo1* apresenta-se como um identificador ou rótulo para o exemplo; ele está diretamente conectado pela relação *Tipo-de* à classe Arco, sobre a qual procuramos formular hipóteses de conceito. Uma coisa importante que pode ser notada é a inexistência de ligações do conceito de Forma a outros pelos atributos *Tipo-de*, *Status* e *Sobre*, conforme sua definição de classe. A explicação é simples; em primeiro lugar, a relação <*Forma* → *Tipo-de* → *Objeto*> pode ser vista como um indicador existencial; ela exprime um nível de abstração tão alto que sua indicação não é necessária (atributo default), pois sua abrangência alcançaria por herança qualquer descrição de objetos físicos. Em segundo lugar, as relações <*Forma* → *Status* → {*Em-pé* / *Deitado*}> e <*Forma* → *Sobre* → {*Formas*}> foram herdadas por Quadrado1, um objeto ligado por especialização à classe Forma. As heranças destas relações são respectivamente <*Quadrado1* → *Status* → *Em-pé*> e <*Quadrado1* → *Sobre* → {*Bloco1*, *Bloco2*}>.

A herança, como sabemos, é realizada pelas operações de generalização e especialização apresentadas na seção 4.4.2.3. Uma característica do S3O é a utilização destas operações através de um controle heurístico, que possibilita ampliar ou diminuir o número de fatos elementares a considerar. Podemos ver que no exemplo corrente, se considerarmos apenas os fatos que referenciam Bloco1 por exemplo, apenas um único exemplo definido na Amostra forneceria fatos válidos; esse problema é denominado "superespecificação".

A classe Objeto possui um nível de abstração muito alto e por isso mesmo sua utilização não é interessante. Portanto, a operação de generalização permite quase sempre ampliar o número de fatos a considerar na amostra. Existe, porém, o perigo de alcançar níveis muito altos de abstração (supergeneralização), gerando fatos com pouca ou nenhuma capacidade de exprimir conceitos. Ob-

serve na Fig. 4.10, o resultado da aplicação de generalizações máximas em cada objeto componente de *Exemplo1*.

No caso de realizarmos sobre os fatos que envolvem os objetos Bloco1... Bloco7, uma generalização para suas classes imediatamente superiores, veremos que todas elas referenciam a mesma classe (Bloco). Aumentando o nível da generalização, podemos chegar a algo como mostra a Fig. 4.10.

Observe no exemplo dado, que as classes generalizados a partir da classe Bloco ocorrem exatamente o mesmo número de vezes na Amostra que esta última. Essa observação é importante para o passo da validação majoritária.

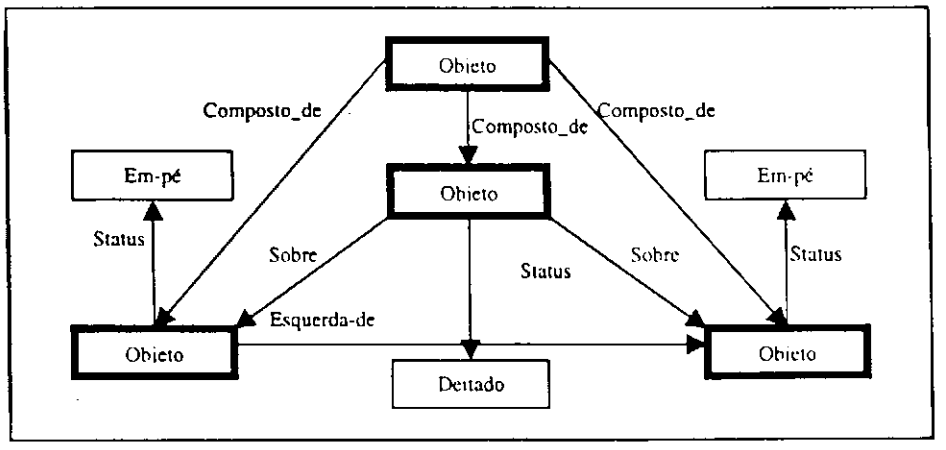


Fig. 4.10 - Resultado de uma Supergeneralização efetuada em *Exemplo1*

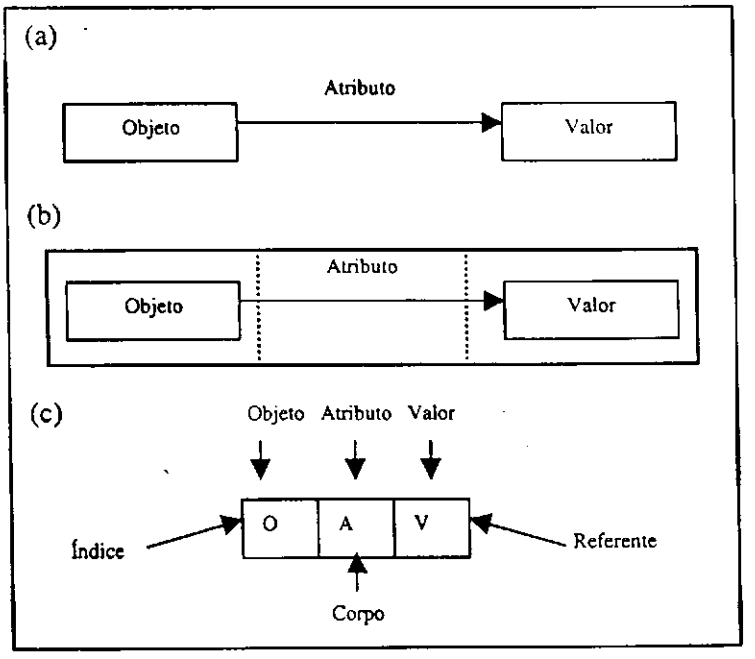


Fig. 4.11 - Descrição para a tupla <O,A,V>

O primeiro passo do S3O consiste simplesmente em identificar todas as partes componentes dos elementos, formando um conjunto de fatos elementares. Um fato elementar é identificado pela estrutura <*O,A,V*> = <*Objeto,Atributo,Valor*> e é o objeto básico manipulado pelo sistema. Observaremos mais adiante que essa estrutura será empregada em estruturas maiores, mas sua representação interna continuará sempre inalterada.

Vamos nesse ponto indicar a segunda forma de descrição mencionada na seção 4.4.2.1. Esta forma é mais interessante para descrever esta estrutura que a vista na Fig. 4.11(a), já que se trata da unidade básica de informação que como já dissemos, permeia todos os seus procedimentos do sistema S3O.

Sem o objetivo de antecipar parte da especificação da estrutura de dados adequada para a implementação do método, consideraremos uma descrição adequada para suportar nosso elemento básico de informação, uma estrutura semelhante à das listas encadeadas **[HOROWITZ 1980]**. As Figs. 4.11(b) e 4.11(c) apresentam o aspecto desta descrição, que é apenas uma transformação do grafo conceitual associado a cada tupla, por meio do encapsulamento das mesmas em uma estrutura composta por três unidades distintas; o *Índice*, o *Corpo* e o *Apontador*. O índice armazena a informação referente ao Objeto, o Corpo armazena o Atributo e o Referente armazena o Valor. Cada uma destas estruturas poderia ligar-se às outras por meio dos Referentes e índices. Os Referentes de uma tupla "apontam" para os índices de outra. Nesse modelo, a partir de um referente, podemos encontrar vários índices, mas a partir de um índice, só podemos referenciar um único apontador; é uma relação 1:N.

O conjunto de fatos elementares é então formado por todas as subestruturas do tipo descrito acima, encontradas na descrição de cada elemento componente da amostra. Para isso, procuram-se todas as tuplas <*O,A,V*> presentes na amostra, onde *O* refere-se à classe do objeto encontrado, *V* refere-se à classe do objeto referenciado (no caso de um valor referencial, ou seja, identificando um outro objeto) ou, caso seja um valor nominal, à classe desse valor (*inteiro*, *real*, *cadeia de caracteres*, ...) e A refere-se à relação entre *O* e *V*, ou seja, a identificação do atributo.

O S3O identifica os fatos elementares a partir da informação do nome da *Classe Principal*, que expressa o conceito a ser aprendido, de um conjunto denominado *Amostra*, que contém os nomes dos exemplos e contra-exemplos de objetos deste conceito, de um conjunto denominado *Classes*, que contém as definições das classes de objetos e finalmente, das instâncias que expressam cada objeto do conjunto Amostra. Podemos observar nas redes conceituais que descrevem o conceito ARCO, que apenas os elementos do conjunto instâncias variam, pois os elementos do conjunto classes são invariáveis e ocorrem em todos os exemplos. Com isso, uma vez que tenhamos

um conjunto "Classes" completo para um determinado domínio, apenas os elementos do conjunto "Instância" serão necessários para definir uma nova rede conceitual.

Para cada um dos fatos elementares encontrados, o S3O gera um conjunto contendo todos os fatos de maior nível de abstração, através de um processo iterativo de generalização. Vamos tomar como exemplo o fato pertencente a *Exemplo1*, apresentado na Fig. 4.12(a). A partir dele, começando pelo primeiro objeto do fato <*Quadrado1*>, o S3O procura nas definições de classe, todas as classes capazes de derivar esse objeto. Podemos identificar nas definições de classe que existem três classes generalizadoras para esse objeto. O S3O constrói, então, três novos fatos, substituindo o objeto inicial por cada uma das classes encontradas. O segundo objeto do fato é <*Bloco1*>, que também possui três classes generalizadoras.

O S3O constrói, em seguida, para cada um dos três fatos gerados para o primeiro objeto, três novos fatos derivados do segundo objeto. Isto é, é realizada uma operação de iteração dentro de outra. Como podemos verificar, quinze fatos são gerados a partir do fato original. Esses fatos são mostrados na Fig. 4.12(b).

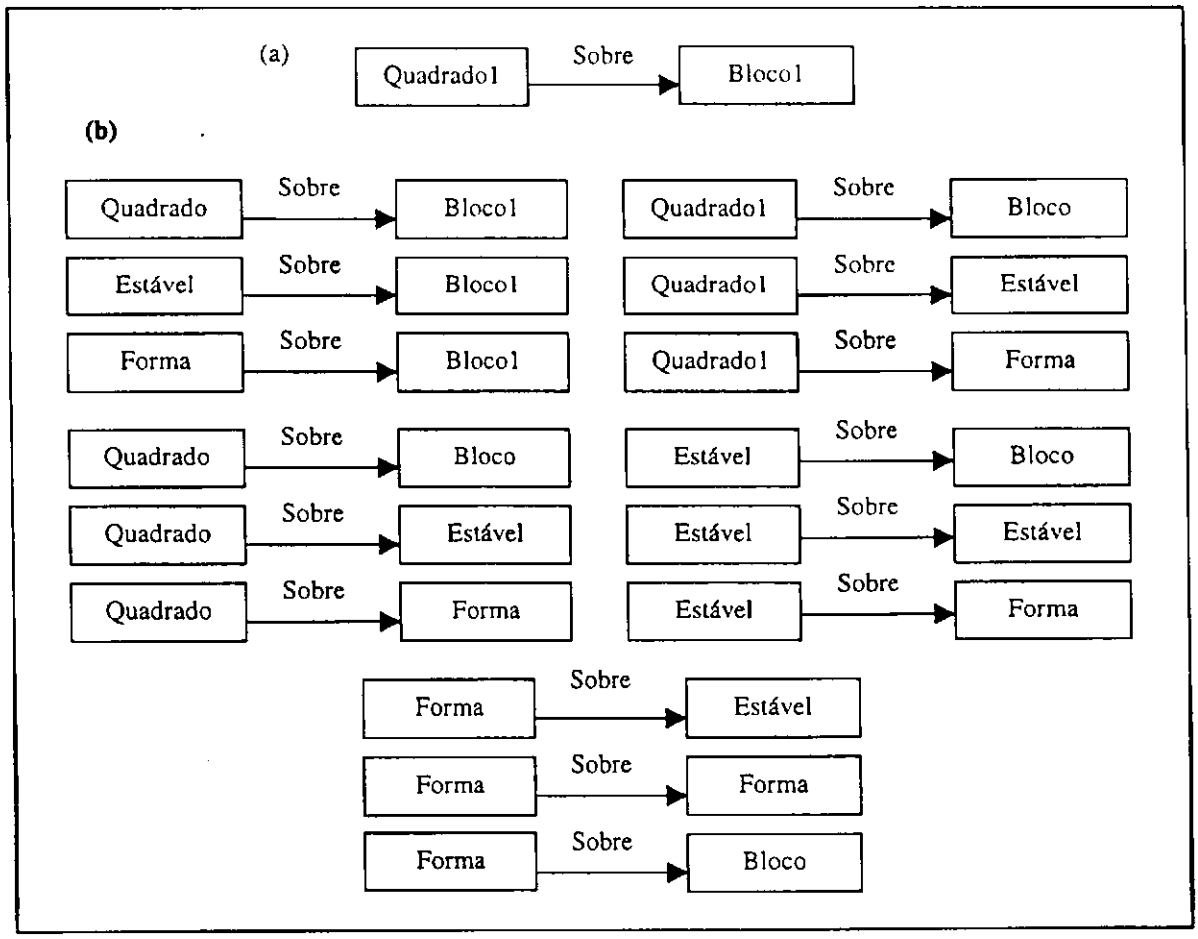


Fig. 4.12 - Geração de Fatos por Generalização.

Além disso, só deverão ser retidos os fatos que são majoritariamente válidos, de acordo com o os limites majoritários estipulados. Esses limites são fornecidos pelo usuário, podem ser alterados como uma forma de melhorar o resultado da aprendizagem e são estabelecidos em termos do percentual de ocorrência dos fatos nos elementos da amostra. O S3O gera o conjunto completo de fatos elementares, mas apenas os fatos que respeitam os limites de validade majoritária são considerados. Para um determinado fato válido, todos os fatos obtidos a partir do mesmo por generalização não são inicialmente considerados. Isto reduz os problema de redundância e de supergeneralização.

Para cada fato encontrado, o S3O estabelece uma relação com o objeto principal e com os fatos definidos em extensão que o mesmo referencia. Com isso, o S3O manipula apenas as definições intensionais, enquanto os fatos definidos em extensão estão permanentemente associados. Quando fatos de objetos diferentes são idênticos aos fatos já identificados, o S3O realiza uma unificação dos mesmos, efetuando novas ligações para referenciar fatos idênticos de objetos diferentes. A quantidade de fatos obtidos pela generalização é função da ordem taxonômica do fato original. Para calcular esta quantidade, podemos efetuar o seguinte cálculo:

Considere n o número de objetos em um fato, $M_{i=1...n}$ o número de classes superiores associadas ao fato i e F o número de fatos derivados por generalização para um fato de n objetos. Temos que:

$$F = \prod(M_i+1) - 1$$

Para o exemplo anterior, $n = 2$, $M_1 = 3$, $M_2 = 3$. Assim,

$$F = [(3+1) * (3+1)]-1 = [4*4]-1 = 16 - 1 = 15$$

Obs: (Esse cálculo não inclui o fato original)

Os enunciados, no caso, os fatos elementares são definidos em dois grupos distintos, conforme visto na seção 4.3.3:

1. **Fatos elementares de intensão**: Isto é, a parte de uma trilha conceitual associada a uma tupla <O,A,V>
2. **Fatos elementares de extensão**: O conjunto das instanciações desta trilha dos objetos que formam a amostra.

Como exemplo para essas definições, a Tab. 4.3 apresenta o conjunto de Fatos Elementares definidos em intensão para os objetos da amostra apresentados nas Figs. 4.9(a,b,c,d); os fatos em negrito são extraídos das instâncias e os restantes são generalizações pelas definições de classes. O primeiro fato elementar apresentado (#01) está relacionado com o subconjunto dos fatos elementa-

res definidos em extensão {<Quadrado1, Status, deitado>} que possui um só componente. Já o fato (#13) está relacionado com o subconjunto dos fatos elementares {<Quadrado1, Status, deitado>,<Retângulo1, Status, deitado>, <Triângulo1, Status, deitado>}.

#	Fato	#	Fato
01	**<Quadrado, Status, deitado>**	17	<Estável,Sobre,Forma>
02	**<Quadrado, Sobre, Bloco>**	18	<Estável,Esquerda-de,Bloco>
03	**<Retângulo, Status, deitado>**	19	<Estável,Esquerda-de,Estável>
04	**<Retângulo, Sobre, Bloco>**	20	<Estável,Esquerda-de,Forma>
05	**<Triângulo, Status, deitado>**	21	<Instável,Status,deitado>
06	**<Triângulo, Sobre, Bloco>**	22	<Instável,Sobre,Bloco>
07	**<Oval, Status, deitado>**	23	<Instáve,Sobre,Estável>
08	**<Oval, Sobre, Bloco>**	24	<Instável,Sobre,Forma>
09	**<Bloco, Status, em-pé>**	25	<Forma,Status,em-pé>
10	**<Bloco,Esquerda-de,Bloco>**	26	<Forma,Status,deitado>
11	<Bloco,esquerda-de,Estável>	27	<Forma,Sobre,Bloco>
12	<Bloco,esquerda-de,Forma>	28	<Forma,Sobre,Estável>
13	<Estável,Status,Deitado>	28	<Forma,Sobre,Forma>
14	<Estável,Status,em-pé>	30	<Forma,Esquerda-de,Bloco>
15	<Estável,Sobre,Bloco>	31	<Forma,Esquerda-de,Estável>
16	<Estável,Sobre,Estável>	32	<Forma,Esquerda-de,Forma>

Tab. 4.3 - Fatos Elementares extraídos da Amostra

Utilizando a segunda forma de descrição apresentada na Fig. 4.11, podemos apresentar a maneira de como o S3O mantém o vínculo entre os conjuntos de intensão e extensão. Visto que poderá existir mais de um fato em intensão idêntico, o S3O elimina as redundâncias mantendo um registro dos objetos onde ocorreu cada fato. Assim, basta manter a associação dos fatos definidos em extensão para cada fato definido em intensão identificado, como mostra a Fig. 4.13.

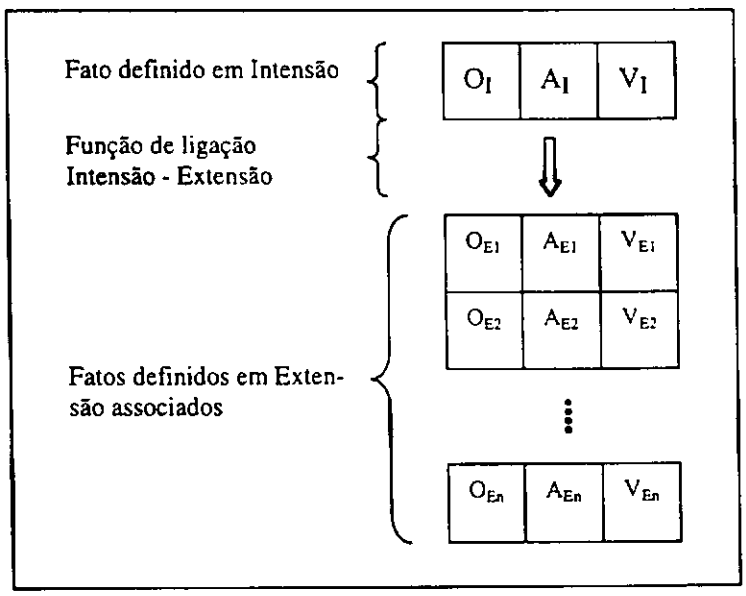


Fig. 4.13 - Ligação dos conjuntos Intensão e Extensão

A partir desse ponto, o S3O deverá deverá construir dois conjuntos distintos:

1. Um conjunto de enunciados majoritariamente válidos (comumente encontrados nos objetos da amostra), chamados *enunciados de assimilação*, como mostrado na Fig. 4.14

2. Um conjunto de enunciados majoritariamente válidos (comumente encontrados nos exemplos do conceito e raramente nos contra-exemplos), chamados *enunciados de discriminação*, como mostrado na Fig. 4.15.

Em suma, sendo dado um conjunto E de objetos estruturados ditos exemplos e C, um outro conjunto de objetos de mesma natureza, ditos contra-exemplos, o problema consiste em encontrar um conjunto de classes de equivalência de subestruturas correspondentes aos objetos da amostra tal que as subestruturas dessas classes sejam fatos majoritariamente válidas sobre a amostra $<E,C>$.

Fatos Elementares de Assimilação Majoritariamente Válidos	**Grafos Conceituais associados**
<Forma, Status, Deitado>	Forma —Status→ Deitado
<Bloco. Status. em-pé>	Bloco —Status→ Em-pé
<Forma, Sobre,Bloco>	Forma —Sobre→ Bloco

Fig. 4.14 - Fatos Elementares de assimilação Majoritariamente Válidos encontrados na Amostra

Fatos Elementares de Discriminação Majoritariamente Válidos	**Grafos Conceituais associados**
<Estável, Status, deitado>	Estável —Status→ Deitado
<Bloco, Status, em-pé>	Bloco —Status→ Em-pé
<Estável, Sobre, Bloco>	Estável —Sobre→ Bloco
<Bloco,esquerda-de,Bloco>	Bloco —Esquerda-de→ Bloco

Fig. 4.15 - Fatos Elementares de Discriminação Majoritariamente Válidos

Uma vez que tenhamos construído esses dois conjuntos, o próximo passo é utilizá-los para a construção de dois outros conjuntos: os Fatos de Assimilação e os Fatos de Discriminação sobre a amostra.

4.6.2 Identificação das Trilhas Conceituais

Os objetos da amostra estão descritos em grafos conceituais, como mostrado na seção 4.5.1. Precisamos agora extrair dos mesmos todas as trilhas conceituais, mantendo sempre o vínculo existente entre cada trilha e o identificador do objeto principal das mesmas. Devemos fazer isso de maneira que as trilhas redundantes sejam eliminadas, assim como fizemos para os fatos elementares, para que na fase de busca de similaridades, duas trilhas idênticas não sejam processadas. Para manter este vínculo, empregamos a mesma maneira que fizemos para manter o vínculo entre os fatos definidos em intensão e os definidos em extensão no caso dos enunciados elementares (Fig. 4.13), já que uma trilha conceitual nada mais é que uma seqüência de fatos elementares.

Considerando a estrutura básica como definimos na seção 4.5.1, um objeto da amostra tem uma descrição semelhante à mostrada na Fig. 4.16. De acordo com a definição de trilhas conceituais apresentadas na seção 4.4.2.5, podemos identificar na estrutura desse exemplo as cinco trilhas conceituais mostradas na Fig. 4.17. A identificação dessas trilhas nos permitirá realizar com maior facilidade, a procura de fatos que ocorrem nos objetos que formam a Amostra. Esses fatos são extraídos utilizando apenas as informações trazidas pela especificação das Instâncias. Como veremos um pouco mais adiante, essas informações são insuficientes para nosso interesse e devem ser completadas com as informações trazidas pelas definições das classes desses objetos.

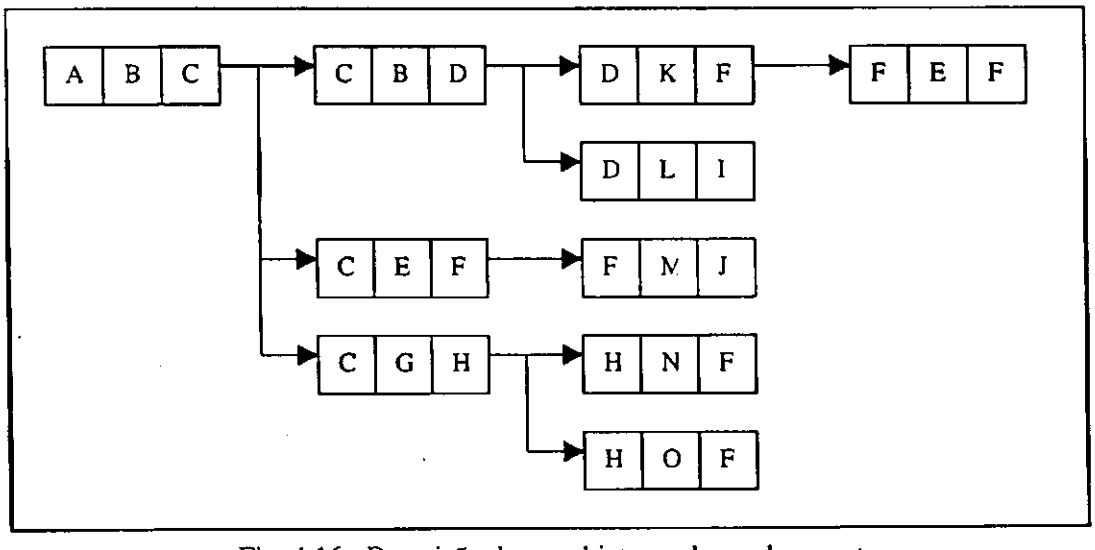


Fig. 4.16 - Descrição de um objeto qualquer da amostra

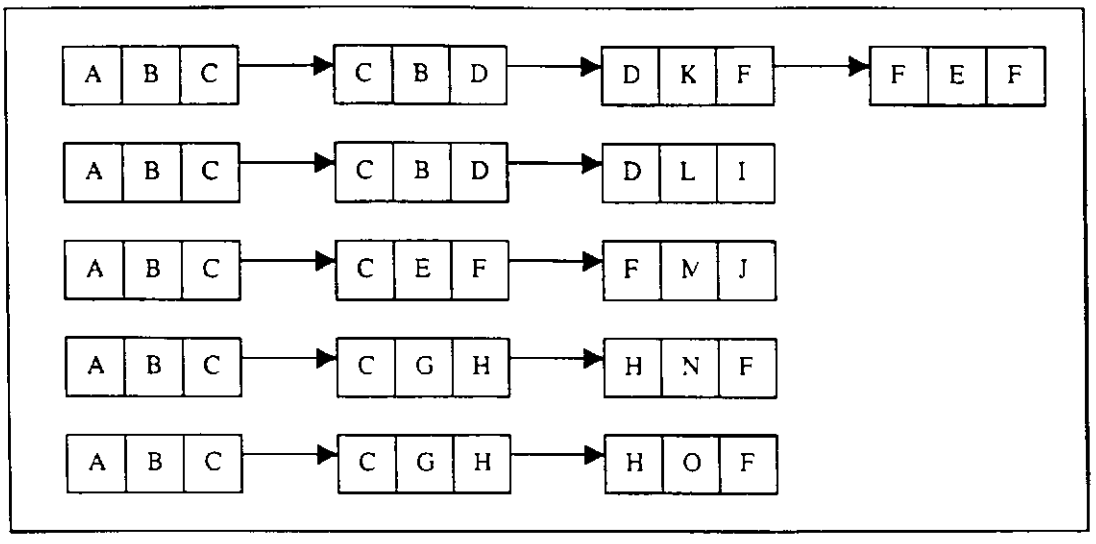


Fig. 4.17 - Trilhas conceituais identificadas

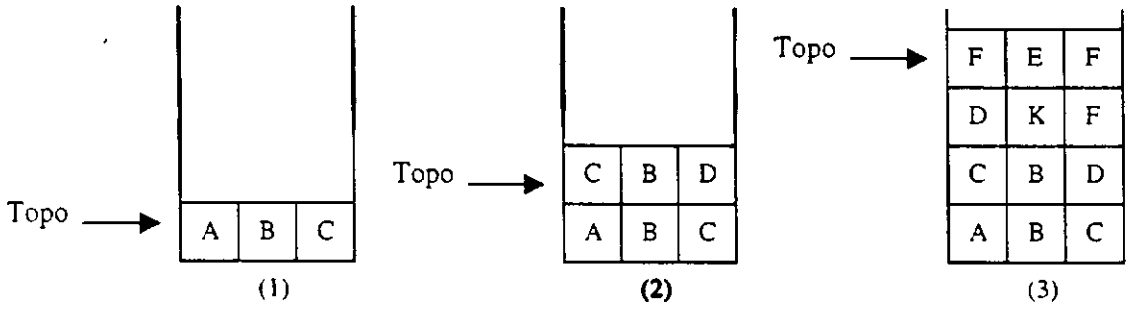
Para efetuar a identificação das trilhas conceituais, definimos para o S3O um algoritmo de caminhamento em árvores baseado no *Caminhamento em Profundidade*. Esse tipo de caminhamento é feito quando se pretende explorar ao máximo determinada ramificação de um grafo.

O algoritmo de caminhamento que definimos é composto pelos seguintes passos:

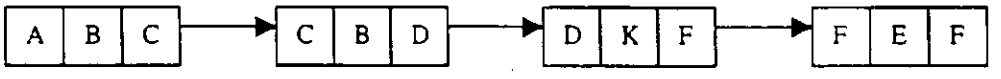
1. Toma-se a tupla que identifica o objeto principal como o vértice inicial;
2. A partir do mesmo, visita-se a primeira tupla apontada, empilhando-a juntamente com o primeiro;
3. Repete-se o procedimento #2 até que não haja mais tuplas apontadas; nesse ponto, a pilha indica a primeira trilha conceitual identificada. Esta trilha deve ser copiada.
4. Iniciamos agora o caminhamento para trás, desempilhando, até encontrar-mos uma tupla que aponte para outra que não esteja na pilha.
5. Visitamos agora essa nova tupla, repetindo os passos 2,3 e 4.
6. O processo se repete até que o algoritmo retorne à primeira tupla empilhada.

Para melhor esclarecer esse algoritmo, vamos considerar por exemplo, a descrição do objeto mostrada na Fig. 4.16 .

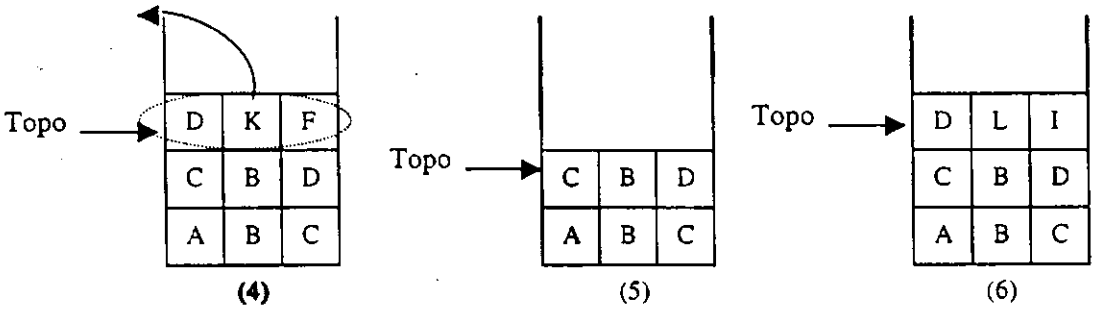
A seqüência de caminhamento inicia-se com a escolha da tupla [A][B][C], que identifica o objeto principal, pois é o vértice inicial (ou raiz); esse vértice é empilhado (passo #1).



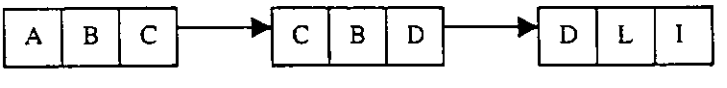
Visitamos a seguir o primeiro vértice apontado por esta tupla, empilhando o mesmo (passo #2). Esse procedimento se repete até que não existam mais tuplas apontadas (passo #3). Nesse ponto, o algoritmo copia as tuplas da pilha, formando assim, a primeira trilha conceitual identificada:



Agora desempilhamos a última tupla inserida na pilha e verificamos se a tupla presente no topo da pilha aponta para alguma outra tupla. Nesse caso, isto não ocorre e a tupla apontada é desempilhada (passo #4).

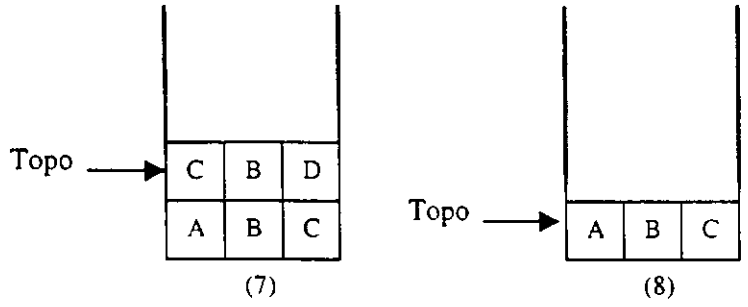


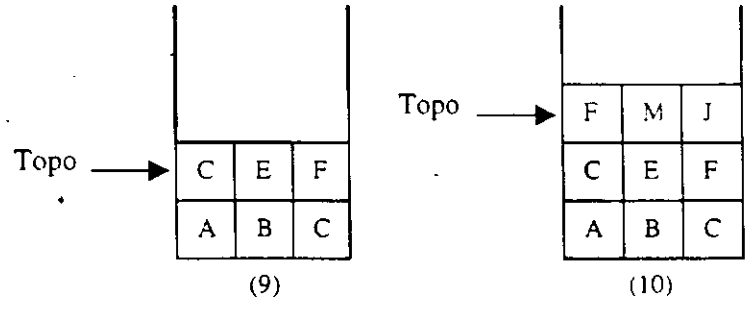
Repetimos o procedimento e verificamos que a tupla [C][B][D] aponta para outra tupla (passo #5). Visitamos a mesma e a empilhamos. A tupla presente no topo não aponta para nenhuma outra; Copiando a pilha, encontramos uma nova trilha conceitual (passo #6).



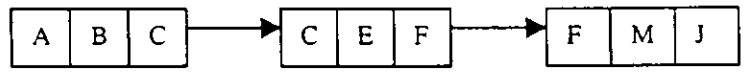
O algoritmo prossegue e desempilha a última tupla inserida na pilha. Verificamos então se a tupla presente no topo da pilha aponta para alguma outra tupla (passo #7). No caso do exemplo, isto não ocorre. Desempilhamos mais uma vez e chegamos à primeira tupla inserida (passo #8); ela aponta para outra tupla. Empilhamos a tupla apontada (passo #9). A nova tupla do topo aponta para

uma outra tupla, que é empilhada (passo #10).

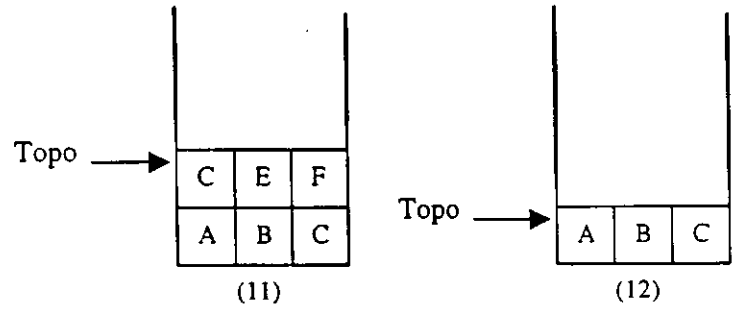


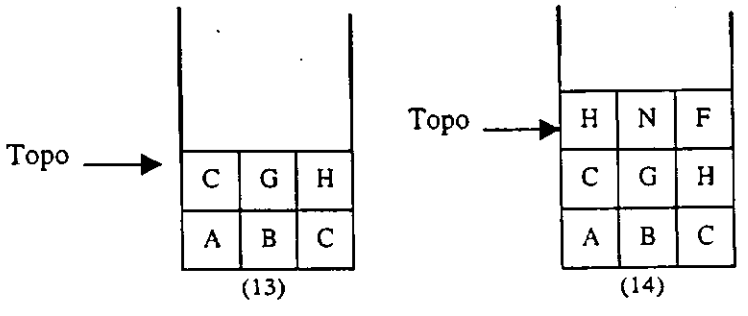


Como a tupla presente no topo não aponta para nenhuma outra tupla, identificamos a terceira trilha conceitual. Copiando a pilha, obtemos:

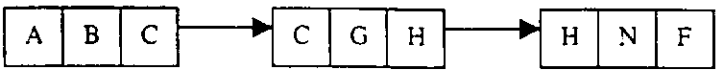


Obtemos como resultado o retorno à tupla inicial. Verificamos que a mesma ainda aponta para a tupla [C][G][H], que aponta para a tupla [H][N][F], que não aponta para nenhuma tupla.

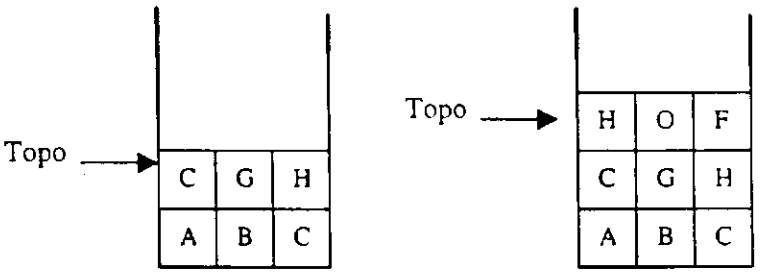




Identificamos a quarta tupla; copiamos a mesma, obtendo:

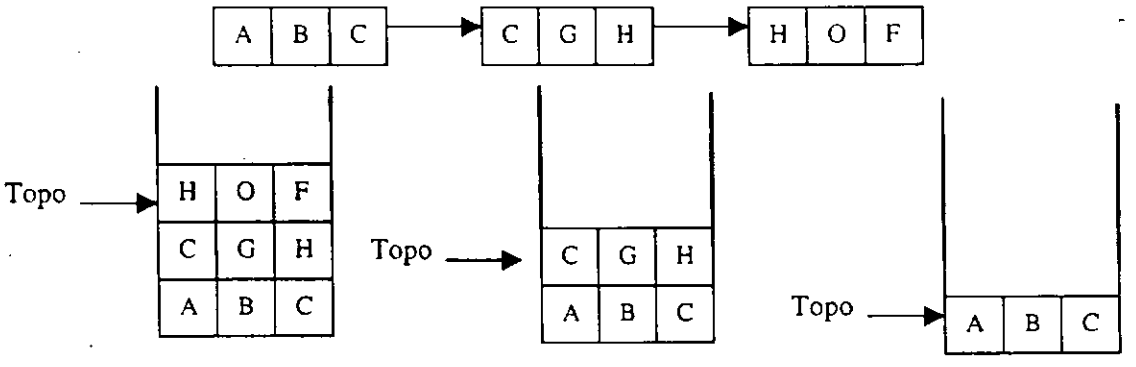


Retornamos à tupla [C][G][H]. Verificamos que a mesma aponta para a tupla [H][O][F] e a empilhamos (passo 16); esta tupla não aponta para nenhuma outra.



(15) (16)

Identificamos a quinta tupla; copiamos a mesma, obtendo:



(17) (18) (19)

Desempilhando, o algoritmo retorna à primeira tupla inserida na pilha. Como não existirá mais nenhuma tupla apontada pela mesma, o algoritmo termina. Nesse ponto, já teremos encontrado cinco trilhas. A Fig. 4.18 mostra a seqüência ou ordem de visitas realizadas pelo algoritmo para o exemplo visto.

A complexidade associada a esse algoritmo para N estruturas de tamanho médio T é $O(2NT-1)$, com exigência de espaço $O(NT)$.

Para cada uma das trilhas conceituais encontradas, o S3O gera a um conjunto contendo todas as trilhas de maior nível de abstração, justamente como gerou os fatos conceituais de maior nível de abstração. Após a geração destas trilhas conceituais, o S3O disporá das seguintes informações:

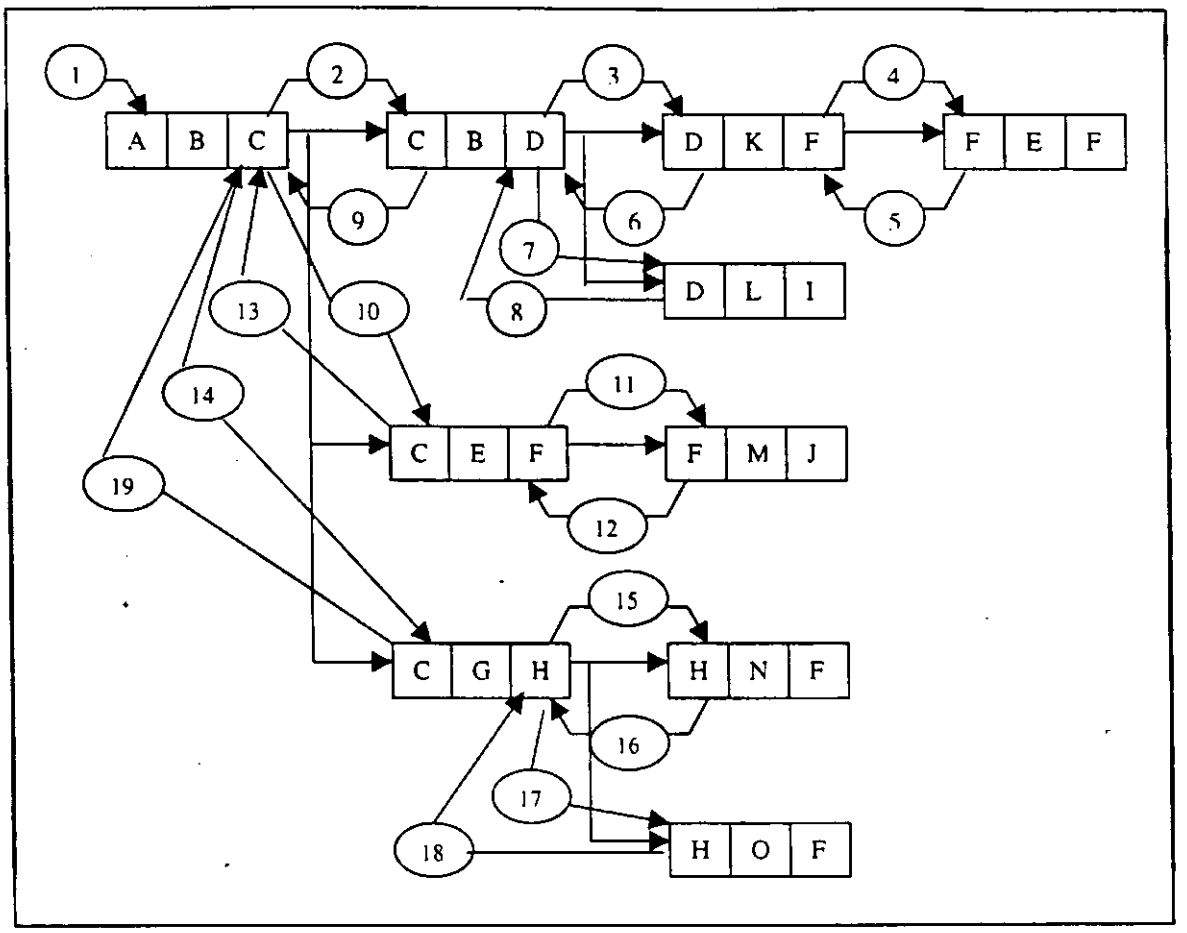


Fig. 4.18 - Ordem de visitação

- O conjunto de Fatos Elementares Majoritariamente Válidos, que inclui os fatos originais das Instâncias dos objetos da Amostra e os objetos de maior nível de abstração (alta generalização) gerados pelas descrições de classes associadas.
- O conjunto das trilhas conceituais que formam os objetos da Amostra, que também incluem os objetos de maior nível de abstração (alta generalização).

4.6.3 A identificação das similaridades

A partir do conjunto de fatos elementares gerado na primeira fase (seção 4.5.1), e dos mesmos tipos de parâmetros, serão executados os seguintes passos:

1. Identificação da classe que representa o objeto composto (Classe Principal),
2. Limites positivo e negativo de validade majoritária,
3. Identificação do tipo de enunciados que serão gerados - assimilação ou discriminação,
4. Tamanhos mínimo e máximo que os fatos gerados deverão ter.

Busca-se a construção de subestruturas de comprimento *i* pela concatenação de duas subestruturas de comprimento *i-2*, como mostra a fig. 4.19.

A construção de um caminho se dá pela busca exaustiva de casamentos possíveis entre trilhas complementares duas a duas, isto é, pares de trilhas ou subtrilhas contendo, na primeira um valor que corresponde a um índice que aponta para a segunda. Definimos esta função como uma fusão de trilhas e a indicamos pelo símbolo (♦); Por exemplo,

<Bloco, Esquerda-de- Bloco> ♦ <Bloco, Status, Em-pé>
➔ <Bloco, Esquerda-de- Bloco, Status, Em-pé>

Iterativamente, a partir do conjunto de fatos elementares, procedemos da seguinte maneira:

1. São construídos enunciados (fatos) de comprimento *i* a partir dos de comprimento *i-1*.
2. Elimina-se todos os enunciados de comprimento *i-1* utilizados para a construção dos de comprimento *i*.
3. O S3O gera inicialmente todo o conjunto de fatos de comprimento *i*, desta maneira, podemos eliminar as redundâncias antes de verificar os limites majoritários, evitando a verificação de fatos repetidos. Nesse ponto, o S3O aciona o algoritmo de casamento de padrões (seção 4.5.4) e encontrará todas as ocorrências desses fatos nas trilhas de comprimento ≥ *i* encontradas (seção 4.5.3).
4. Retêm-se apenas os caminhos majoritariamente válidos em relação à amostra e cujo comprimento estão dentro dos limites estipulados.
5. Esta operação se repete para cada *i*, até que o conjunto gerado seja vazio.

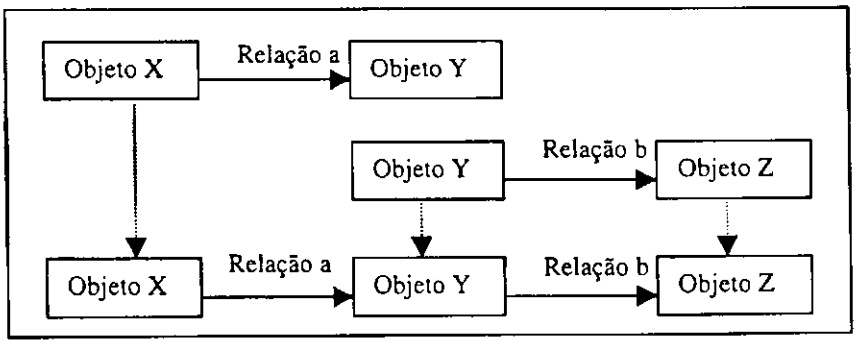


Fig. 4. 19 - Concatenação de Fatos para n=1

A concatenação de dois elementos, como vista nesse parágrafo, consiste em estabelecer uma simples ligação entre o referente de um elemento e o índice de outro, com mesmo valor. Por exemplo, o fato concatenado <a,b,c,d,e,f,g> teria estrutura de listas como mostra a fig. 4.20.

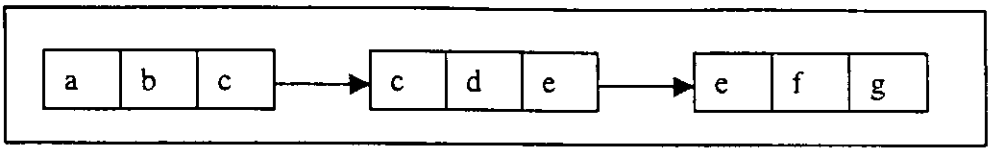


Fig. 4.20 - Concatenação pela ligação referente-índice

4.6.4 Casamento de Padrões (Pattern Matching)

O casamento de padrões é um problema importante, que ocorre nas mais variadas áreas no processamento de informações. Dentre as situações pertinentes à área da computação, podemos citar o processamento de texto e a computação gráfica. Na IA, existe toda uma área de estudos de problemas desta natureza, a Percepção de Padrões, que se trata em resumo, de implementar em um sistema artificial a capacidade de encontrar uma descrição útil para algo descrito de maneira muito complexa (ou com pouca utilidade). Quando se procura encontrar uma descrição mais simples de alguma coisa ou objeto, é quase sempre possível utilizar algumas propriedades do mesmo, como a forma, o desenho ou a regularidade. Se esta propriedade existe e está disponível, então a descrição complexa do objeto é dita ser um exemplo de um padrão [JACKSON 1985].

Talvez as formas mais simples de problemas desse tipo ocorram em processamento de textos, quando se procura, por exemplo, localizar a ocorrência de uma determinada cadeia de caracteres específica dentro de uma determinada área de texto.

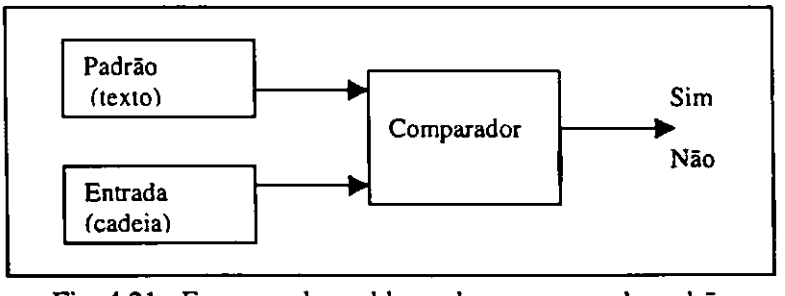


Fig. 4.21 - Esquema do problema de casamento de padrões

De maneira geral, um problema de casamento de padrões segue o esquema da Fig. 4.21. Nesse exemplo, instanciamos o problema para o caso da pesquisa de cadeias de caracteres em texto. A saída informa se a cadeia de caracteres ocorre ou não no texto.

De maneira geral, o problema de casamento de padrões envolve dois conjuntos de elementos da mesma natureza e o problema consiste em descobrir se um deles pertence ao outro. No caso simples do processador de texto, podemos identificar que em qualquer situação o problema se resume a testar a ocorrência de uma cadeia de n caracteres num texto de m caracteres. A solução mais

óbvia para resolver esse problema nos dá uma idéia da complexidade associada ao mesmo; trata-se do método da força-bruta, um algoritmo que varre toda a cadeia de caracteres a cada vez que se desloca uma posição no texto - A Fig. 4.22 mostra a comparação da seqüência S no texto T a partir da Segunda posição do texto. Esse algoritmo resolve o problema no tempo máximo $O(mn)$ e no espaço de $O(m)$.

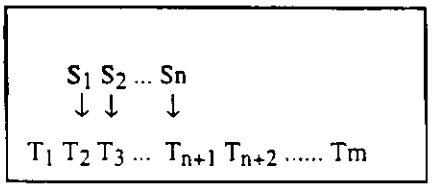


Fig. 4.22 - Busca pelo método da força-bruta

Nesse passo do nosso método, precisamos utilizar um algoritmo de casamento de padrões para encontrar dentro das descrições de cada elemento da amostra as trilhas conceituais geradas a partir dos fatos elementares. Para otimizar a solução desse problema, evitando ao máximo a perda de eficiência resultante da aplicação do método da força bruta, utilizaremos um método conhecido de casamento de caracteres, adaptando o mesmo para trabalhar com a estrutura elementar da nossa informação, a tupla <O,A,V>. Trata-se do algoritmo de casamento de cadeias caracteres de Boyer-Moore. Esse algoritmo é até hoje, o mais eficiente para resolver esse tipo de problema; a complexidade do problema decai vertiginosamente em relação ao método da força bruta. O algoritmo de Boyer-Moore resolve o problema de casamento de cadeia de caracteres em tempo não superior a $O(m+n)$ num espaço $O(m)$ **[LEEUWEN 1990]**.

4.6.4.1 O Algoritmo de Boyer-Moore

O algoritmo de Boyer-Moore é uma variação interessante do famoso algoritmo de Knuth-Morris-Pratt **[LEEUWEN 1990]**. Esse método é baseado na idéia de que um algoritmo pode ganhar mais informação casando padrões a partir da direita que a partir da esquerda. O algoritmo procura pela primeira ocorrência de um determinado padrão no texto. Durante a busca, os caracteres são varridos para um casamento com o último caractere do padrão. A informação obtida com o início do casamento pelo último caractere pode permitir ao algoritmo deslocar o padrão procurado em várias posições no texto. O algoritmo utiliza duas funções heurísticas para alcançar esta tarefa - as funções *Delta1* e *Delta2*.

- A função **Delta1** (Δ_1) verifica-se quando um erro ocorre (os objetos comparados são diferentes). A função utiliza a informação sobre a posição onde o erro ocorreu no padrão, para produzir um novo deslocamento. O número de deslocamentos depende do caractere

obtido do texto. Se o caractere não ocorre no padrão, Δ_1 é, então, o comprimento do padrão. Se o caractere ocorre no padrão, então podemos deslocar o padrão em um número de posições igual a Δ_1, sem procurar por outros casamentos.

- A função **Delta2** (Δ_2) é uma função da posição do caractere do padrão onde ocorreu o erro. Esta função utiliza a informação dos últimos caracteres do padrão casados com o do texto, da direita para a esquerda (subpadrão) e ainda a informação da repetição desse subpadrão mais à direita no texto.

O algoritmo procurará o resultado de qualquer uma destas funções que em sendo aplicada, resulte no maior deslocamento do apontador da função de comparação. O exemplo apresentado na seção a seguir, ilustra a aplicação de Δ_1 e Δ_2.

4.6.4.2 O Algoritmo de Boyer-Moore Modificado

Considere para o nosso caso, as seguintes definições:

- *FATO*: é um fato pertencente ao conjunto E_i, cuja ocorrência na descrição de um determinado objeto se procura verificar. Seu tamanho deve ser menor ou igual ao comprimento máximo pré-estabelecido para fatos.
- *SUBFATO*: é a parte de *FATO* mais à direita, encontrada em *TRILHA* até então.
- *TRILHA*: é a trilha (ou subtrilha) conceitual na qual o algoritmo procura pela ocorrência do *FATO*.
- *OBJETO*: é o elemento de *TRILHA* sendo apontado pelo algoritmo.
- *COMPARAÇÃO*: é uma função binária que compara os valores de *TRILHA* e *FATO* apontados pela mesma, retornando dois resultados distintos possíveis:

Verdadeiro (=) ou **Falso** (≠)

As duas funções heurísticas do algoritmo nos permite enumerar três observações ou situações diferentes:

- **Observação #1**: Se soubermos que *OBJETO* não ocorre em *FATO*, então não existe a necessidade de considerar a possibilidade de uma ocorrência de *FATO* iniciando na posição 1 de *TRILHA*, que é indicada pela função *TAMANHO(FATO)*.
- **Observação # 2**: Se a ocorrência mais à direita de *OBJETO* em *FATO* localiza-se Δ_1 po-

sições mais à direita de *FATO*, então podemos deslizar *FATO* de um número de posições igual a Δ_1, sem procurar por casamentos. Se deslizarmos *FATO* menos que Δ_1 posições, então o caractere na cadeia não estará alinhado com um caractere que possa casar.

- **Observação # 3**: Se *OBJETO* ocorre na posição mais à direita de *FATO*, então devemos checar se o *OBJETO* da posição anterior na *TRILHA* casa com o da segunda posição mais à direita de *FATO*. Isto pode ser considerado de duas maneiras: 3(a) e 3(b).

- **Observação 3(a)**: Baseado na observação 3, quando a função *COMPARAÇÃO* retorna (≠), ou seja, encontra um erro na verificação da igualdade, queremos deslocar *FATO* de k posições, de maneira que o objeto conhecido *OBJETO*, que encontramos casado com o mais à direita de *FATO* na primeira varredura, alinhe-se com o mais à direita de *TRILHA*. A distância k depende de onde *OBJETO* ocorra em *FATO*. Se a ocorrência mais à direita de *OBJETO* em *FATO* está à esquerda do erro, *FATO* será deslocado para a frente em um número de posições igual a k, com $k = \Delta_1$ (*FATO*) - *TAMANHO (SUBFATO)*.

- **Observação 3(b)**: Desde que os próximos m caracteres de *FATO* coincidem com os m caracteres finais de *TRILHA* (*SUBFATOS* de tamanho m). Usamos *SUBFATO* para deslocar *FATO* pela *TRILHA* em um número de posições igual a k. Após deslocar *FATO* de k posições, queremos inspecionar o caractere de *TRILHA* alinhado com o último caractere de *FATO*. O resultado da distância é Δ_2

Uma trilha (ou sub-trilha) conceitual é tratada como uma sucessão ou encadeamento de tuplas <O,A,V>; considerando a descrição estrutural vista em 4.4.2.5, uma trilha conceitual teria o aspecto mostrado na Fig. 4.23.

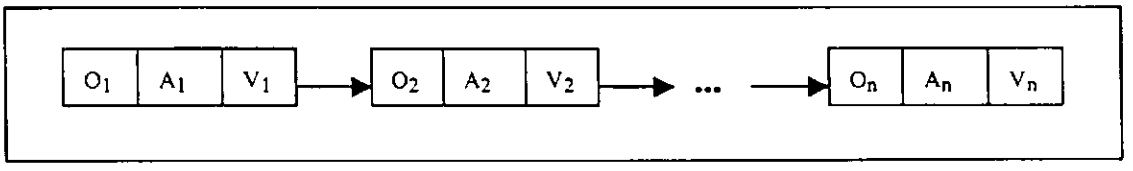


Fig. 4.23 - Elementos do S3O ligados

Nosso problema pode ser reescrito da seguinte forma: como procurar numa trilha (ou sub-trilha) conceitual a ocorrência de uma outra estrutura de menor ou igual tamanho - os fatos resultantes das concatenações (vista aqui como ligações) dos fatos elementares majoritariamente válidos, para formar o conjunto dos fatos majoritariamente válidos sobre a amostra?

Para simplificar nossa explicação, vamos recorrer a um exemplo. Para isso, vamos considerar que as ligações entre os elementos da lista estão representadas não por setas, mas pela concate-

nação dos elementos como mostrado na Fig. 4.23. A lista vista acima seria representada como mostra a Fig. 4.24.

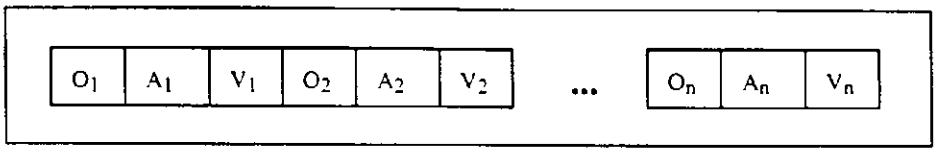


Fig. 4.24 - Representação dos elementos sem ligações explícitas

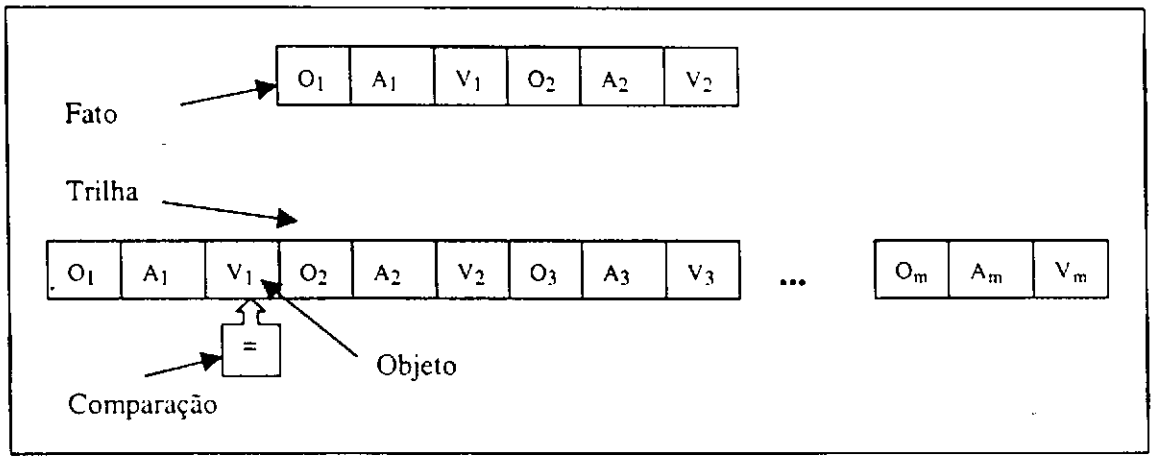


Fig. 4.25 - Elementos do algoritmo de busca de similaridades

Com isso, os dados FATO, TRILHA, OBJETO e COMPARAÇÃO teriam o aspecto mostrado na Fig. 4.25. Na nossa representação, supomos que a seqüência procurada (FATO) desliza para a esquerda ou para a direita sobre a estrutura onde se procura sua ocorrência (TRILHA). Supomos ainda a TRILHA como uma estrutura fixa ou imóvel e o apontador da função COMPARAÇÃO também móvel, também deslizando para ambos os lados.

Para o exemplo, representaremos cada um dos elementos básicos dos objetos por um caractere, como por exemplo: $A = <O_1,A_1,V_1>$.

Supondo que desejamos encontrar a ocorrência do fato ABCCDE na subárvore conceitual descrita por ABCCBEEDAABCCDEEDCCBI temos a seguinte estrutura de dados:

Fato ($n = 6$)

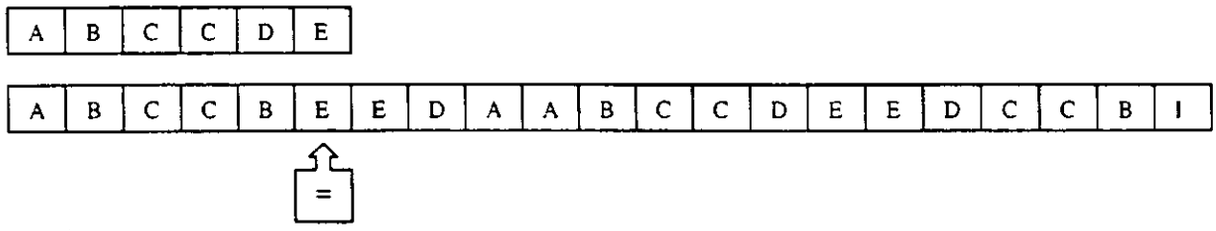
A	B	C	C	D	E

Trilha ($m = 21$)

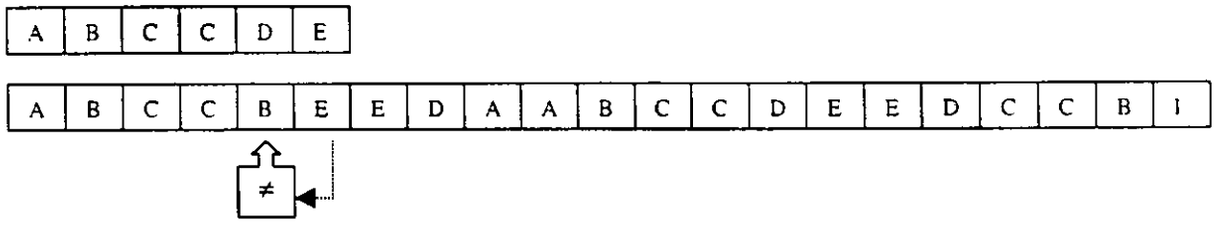
A	B	C	C	B	E	E	D	A	A	B	C	C	D	E	E	D	C	C	B	I

O procedimentoé o seguinte. De início, a seqüência procurada (Fato) é igualada à subárvore

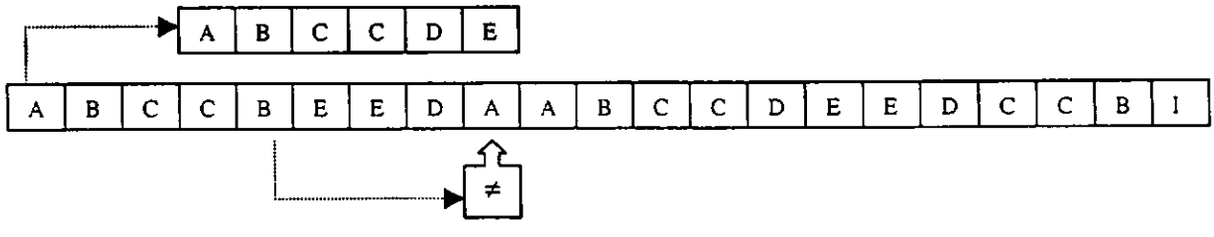
conceitual (Trilha). Ambas são alinhadas pela esquerda e a pesquisa é iniciada pela posição mais à direita. A seta indica a posição correntemente referenciada na Trilha, onde está o padrão (Objeto) da trilha que será comparado pela função Comparação exatamente alinhado com o mesmo.



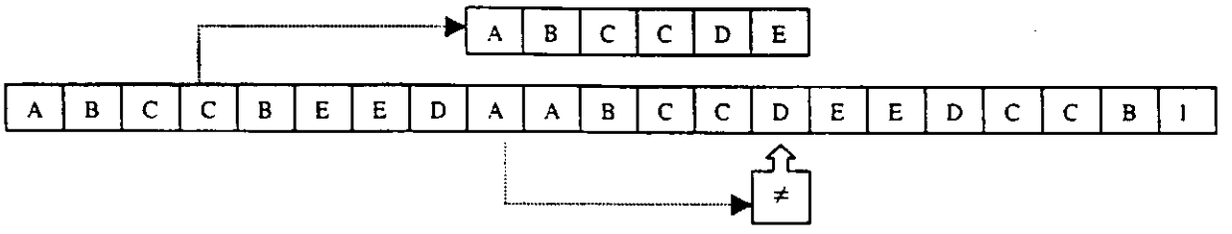
Encontramos de início um casamento entre o 6° elemento da Trilha e o 6° elemento do Fato. $[E]_T = [E]_F$. Com isso, mantemos a posição do Fato e movemos o apontador uma posição para a esquerda, afim de comparar os elementos da posição anterior, já que o número de casamentos encontrados é menor que n (o tamanho do Fato).



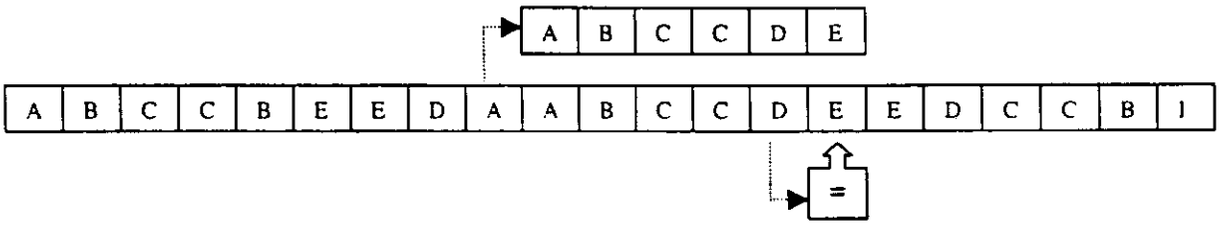
Agora a função de comparação retorna um erro na tentativa de casamento entre o 5° elemento da Trilha e o 5° elemento do Fato. $[E]_T \neq [E]_F$. Notemos que $[E]_T$ ocorre na 2ª posição do Padrão. Pela observação 2, deslizamos o Padrão sobre a Trilha 3 posições, até que $[E]_T$ e $[E]_F$ estejam coincidentes e novamente deslocamos o apontador para onde se localiza elemento mais à direita do Fato.



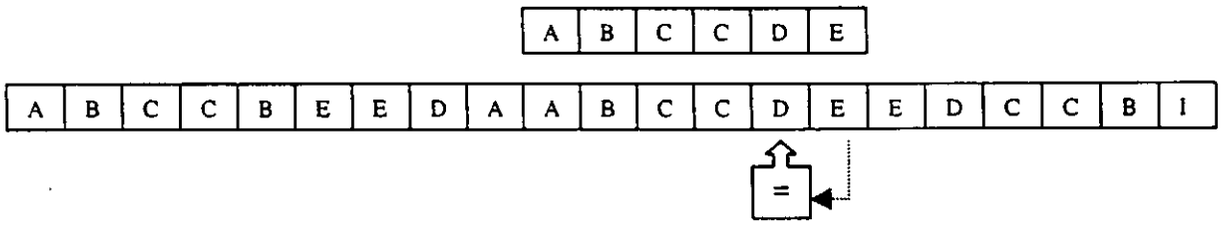
A função de comparação retorna um novo erro na tentativa de casamento entre o 9° elemento da Trilha e o 6° elemento do Fato. $[E]_T \neq [E]_F$. Novamente, o elemento $[E]_T$ ocorre no Fato, desta vez na 1ª posição. Pela observação 2, deslizamos o Fato sobre a Trilha 5 posições, até que $[E]_T$ e $[E]_F$ estejam coincidentes e novamente deslocamos o apontador para onde se localiza elemento mais a direita do Fato.



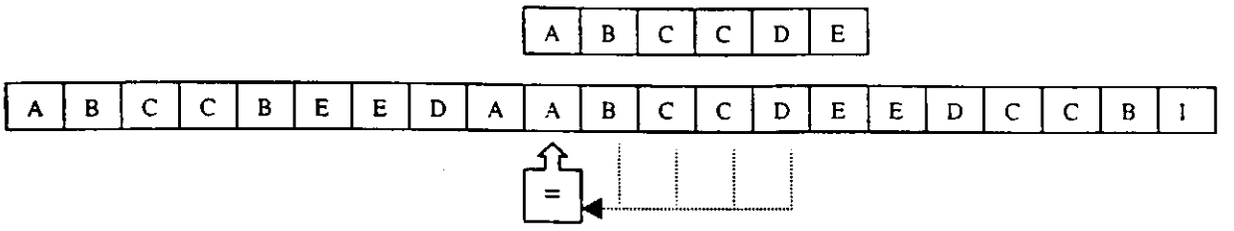
A função de comparação retorna um novo erro na tentativa de casamento entre o 14° elemento da Trilha e o 6° elemento do Fato. $[E]_T \neq [E]_F$. Novamente, o elemento $[E]_T$ ocorre no Fato, desta vez na 5ª posição. Pela observação 2, deslizamos o Fato sobre a Trilha 1 posição, até que $[E]_T$ e $[E]_F$ estejam coincidentes e novamente deslocamos o apontador para onde se localiza elemento mais a direita do Fato.



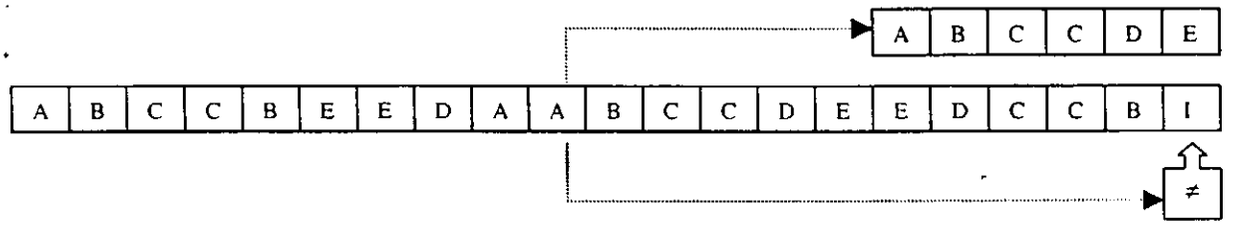
Encontramos um casamento entre o 15° elemento da Trilha e o 6° elemento do Fato. $[E]_T = [E]_F$. Com isso, mantemos a posição do Fato e movemos o apontador uma posição para a esquerda, a fim de comparar os elementos da posição anterior, já que o número de casamentos encontrados é menor que n.



Encontramos um casamento entre o 14° elemento da Trilha e o 5° elemento do Fato. $[E]_T = [E]_F$. Com isso, mantemos a posição do Fato e movemos o apontador uma posição para a esquerda, afim de comparar os elementos da posição anterior, já que o número de casamentos encontrados ainda é menor que n. Esse procedimento continua, até que o encontra o casamento entre o 11° elemento da Trilha e o 2° elemento do Fato. $[E]_T = [E]_F$. Finalmente, encontramos a ocorrência do Fato na Trilha!



Observe porém que o apontador está na 11ª posição da Trilha. Calculando o número de elementos não comparados da Trilha, temos: m - (11+ (n+1)) = 6. Isto indica que ainda resta um número de elementos na Trilha igual ao tamanho do Fato. Como numa trilha uma Prova pode ocorrer mais de uma vez, o algoritmo deve prosseguir. Deslocamos o apontador para a posição (11+2(n-1)) da Trilha e deslizamos a Prova de n posições para a direita. A busca de Fatos em Trilha conceituais realizada pelo S3O percorre todo o espaço de cada trilha, isto é, todos os fatos que ocorrem nas trilha são encontrados, mesmo que se repitam várias vezes.



A função de comparação retorna um novo erro na tentativa de casamento entre o 21° elemento da Trilha e o 6° elemento do Fato. $[E]_T \neq [E]_F$. Agora, não existe um número suficiente de elementos na Trilha para que O Fato possa deslizar sobre a mesma. Assim, o algoritmo finaliza.

Dentre as modificações que fizemos no algoritmo de Boyer-Moore original, uma das mais importantes é o fator de multiplicação do deslocamento do Fato. Enquanto o algoritmo original promove o deslocamento do Padrão uma posição por vez, no nosso algoritmo, tivemos que multiplicar esse deslocamento por três, devido à estrutura básica da nossa informação (a tupla <O,A,V>), que impõe a necessidade de casar os elementos de cada tupla emparelhados. Além do mais, vale lembrar que cada um dos elementos da tupla representa um identificador, cuja estrutura de dados é uma cadeia de caracteres; o que representa uma rotina de comparação de cadeias que pode ser feita com base no valor do código ASCII, em cada posição apontada. O algoritmo modificado de Boyer-Moore resolve o nosso problema de casamento de cadeia de caracteres em tempo não superior a O(m/3+n) num espaço O(m).

Os fatos de mesmo comprimento encontrados são agrupados em conjuntos e o S3O realiza a verificação dos limites de validade majoritária. Os limites são expressos em termos do percentual de ocorrência dos fatos sobre os elementos da amostra. Para o nosso exemplo, vamos estabelecer os seguintes limites:

- Limite positivo de validade majoritária: 51%

- Limite negativo de validade majoritária: Não vamos especificar

Devido ao tamanho reduzido da amostra, não vamos especificar o filtro; isto significa que não haverá restrições sobre os valores dos atributos.

Após a realização do casamento de padrões, o S3O terá encontrado os Fatos Majoritariamente Válidos apresentados nas Figs. 4.26 e 4.27.

Esses dados são mostrados na Tabela 4.4 na forma de tuplas <O,A,V>, acompanhados dos seguintes dados: índice ou contador de fatos encontrados (#), indicador do tipo de interpretação (A/D), número de elementos da Amostra onde ocorre o Fato, número de ocorrência do Fato em Amostra, nível de Profundidade do Fato e a indicação se o fato é ou não maximal (O S3O separa desses fatos aqueles que são maximamente especificados, isto é, elimina os fatos que são constituintes de outros fatos). Na fig. 4.28, é mostrada a árvore de composição desses fatos, onde as folhas (em destaque) são os fatos separados.

Neste ponto, o S3O encontra seu objetivo; o processo de aprendizagem realizado até então foi suficiente para definir os fatos que construirão as hipóteses sobre o conceito que se deseja aprender. Como informado no início do capítulo, será informado, sem maiores detalhes, como o processo de aprendizagem realizado pelo SAID segue adiante.

4.7 A Construção do Objeto Generalizante

Os fatos encontrados pelo S3O devem agora ser agregados numa estrutura que permita definir um objeto generalizante. Tomando como referência as trilhas conceituais de cada exemplo da amostra, checam-se a ocorrência dos fatos.

Considere os quatro exemplos, descritos numa matriz, onde as linhas representam os principais vértices conceituais e as colunas, os fatos encontrados; o número de ocorrência do fato no vértice é inscrito na interseção com a trilha (Fig. 4.29):

O somatório do número de ocorrências de cada fato está incluído nesta descrição. Analisando essas tabelas, chegamos ao seguinte resultado:

- Apenas o fato #12 é válido em todos os exemplos. Ele é o único fato máximo de assimilação encontrado.
- Os fatos 7, 14, 17 e 18 são válidos para os três exemplos que não apresentam ruído. Eles são os fatos máximos de discriminação encontrados.

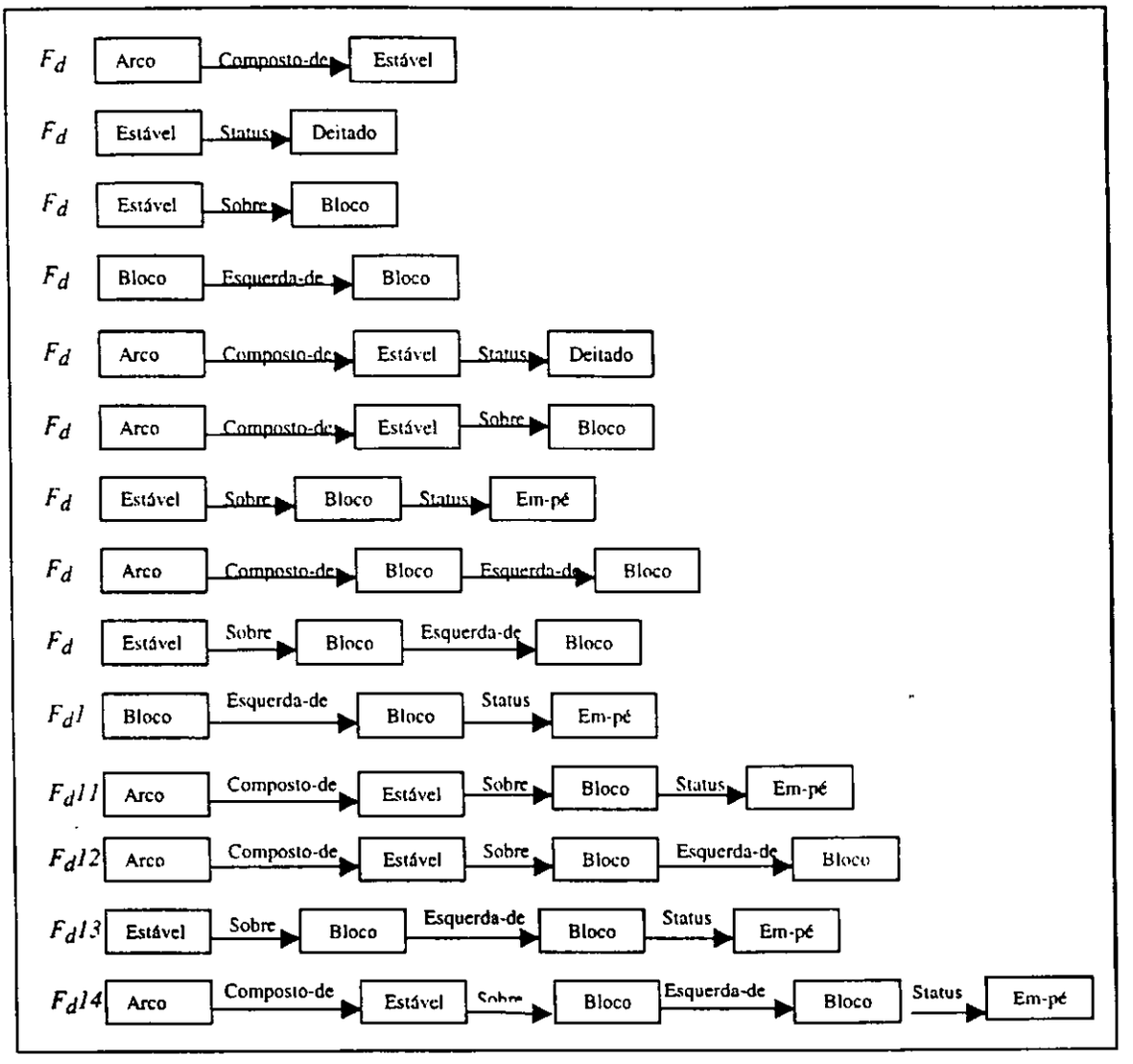


Fig. 4.26 - FMVs de Discriminação

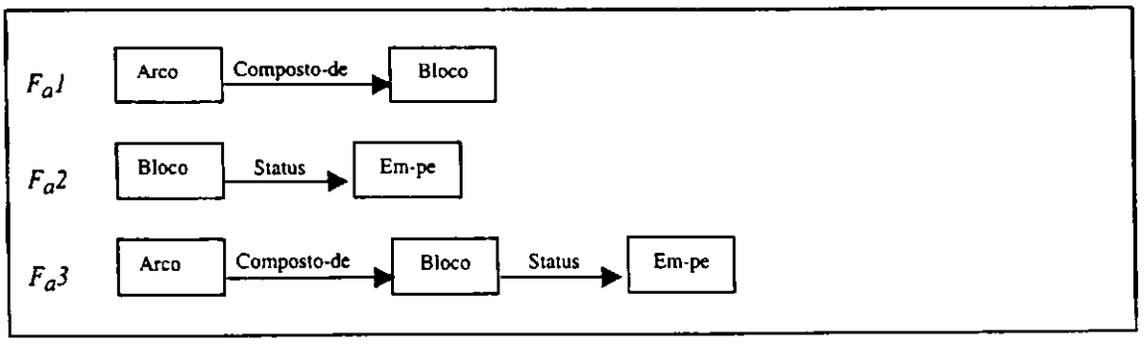


Fig. 4.27 - FMVs de Assimilação

#	FMV	A/D	NO	NF	NP	M
1	<ARCO,COMPOSTO-DE,ESTÁVEL>	D	9	3	1	N
2	<ESTÁVEL,STATUS,DEITADO>	D	3	3	1	N
3	<ESTÁVEL,SOBRE,BLOCO>	D	6	3	1	N
4	<ARCO,COMPOSTO-DE,BLOCO>	A	7	4	1	N
5	<BLOCO,STATUS,EM-PE>	A	7	4	1	N
6	<BLOCO,ESQUERDA-DE,BLOCO>	D	3	3	1	N
7	**<ARCO,COMPOSTO-DE,ESTÁVEL><ESTÁVEL,STATUS,DEITADO>**	**D**	**3**	**3**	**2**	**S**
8	<ARCO,COMPOSTO-DE,ESTÁVEL><ESTÁVEL,SOBRE,BLOCO>	D	6	3	2	N
9	<ESTÁVEL,SOBRE,BLOCO><BLOCO,STATUS,EM-PE>	D	6	3	2	N
10	<ARCO,COMPOSTO-DE,BLOCO><BLOCO,ESQUERDA-DE,BLOCO>	D	3	3	2	N
11	<ESTÁVEL,SOBRE,BLOCO><BLOCO,ESQUERDA-DE,BLOCO>	D	3	3	2	N
12	**<ARCO,COMPOSTO-DE,BLOCO><BLOCO,STATUS,EM-PE>**	**A**	**7**	**4**	**2**	**S**
13	<BLOCO,ESQUERDA-DE,BLOCO><BLOCO,STATUS,EM-PE>	D	3	3	2	N
14	**<ARCO,COMPOSTO-DE,ESTÁVEL><ESTÁVEL,SOBRE,BLOCO><BLOCO,STATUS,EM-PE>**	**D**	**6**	**3**	**3**	**S**
15	<ARCO,COMPOSTO-DE,ESTÁVEL><ESTÁVEL,SOBRE,BLOCO><BLOCO,ESQUERDA-DE,BLOCO>	D	3	3	3	N
16	<ESTÁVEL,SOBRE,BLOCO><BLOCO,ESQUERDA-DE,BLOCO><BLOCO,STATUS,EM-PE>	D	3	3	3	N
17	**<ARCO,COMPOSTO-DE,BLOCO><BLOCO,ESQUERDA-DE,BLOCO><BLOCO,STATUS,EM-PE>**	**D**	**3**	**3**	**3**	**S**
18	**<ARCO,COMPOSTO-DE,ESTÁVEL><ESTÁVEL,SOBRE,BLOCO><BLOCO,ESQUERDA-DE,BLOCO><BLOCO,STATUS,EM-PE>**	**D**	**3**	**3**	**4**	**S**

Convenções:
#: N° do Fato
FMV: Fato Majoritariamente Válido
A/D: [A] Assimilação ou [D] Descrição
NO: N° de elementos da Amostra onde ocorre o Fato.
NF: N° de ocorrência do Fato em Amostra
NP: Nível de Profundidade do Fato.
M: Fato maximal

Tab. 4.4 - FMVs sobre a Amostra

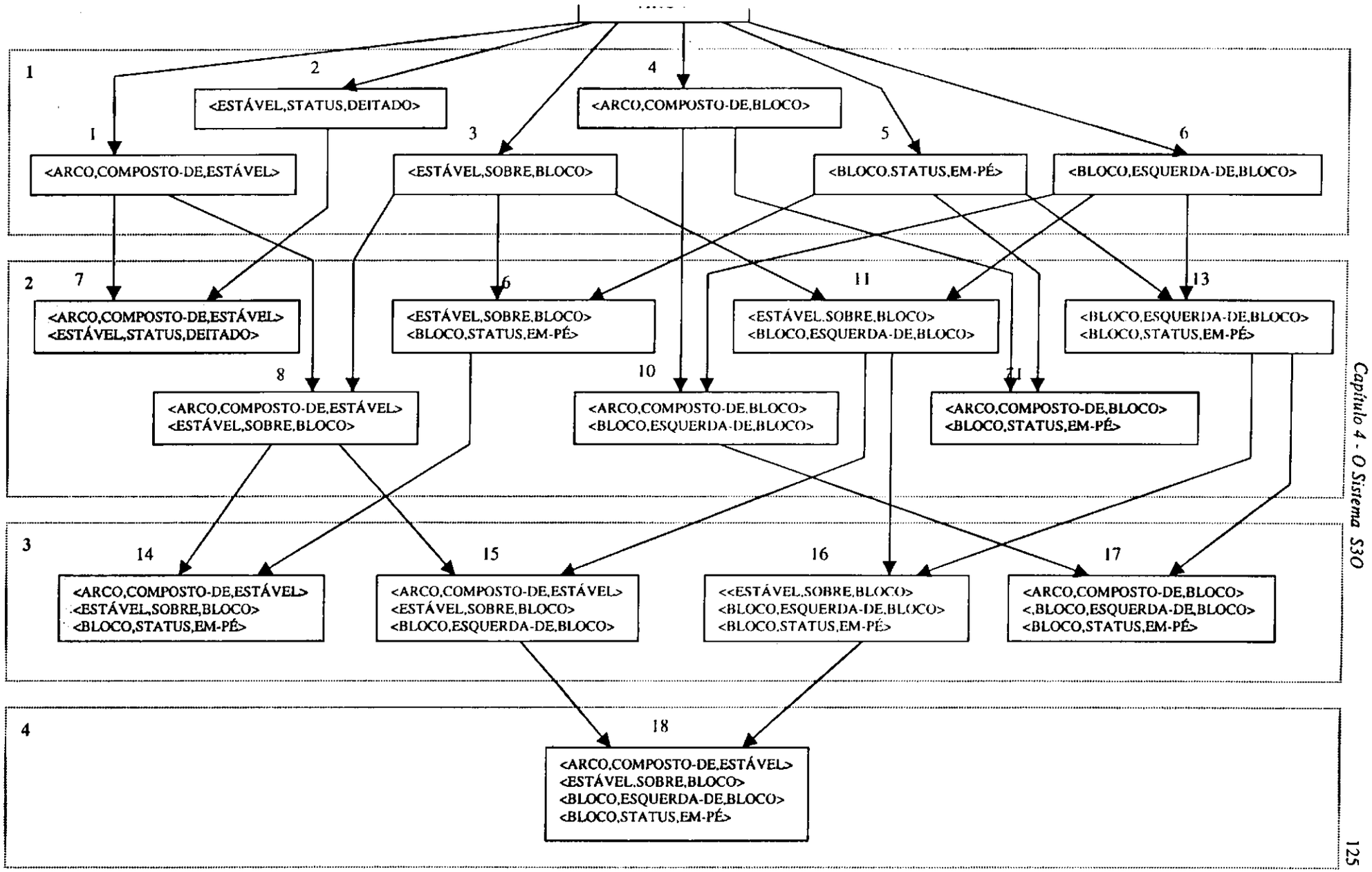


Fig. 4.28 - Árvore de FMVs, com fatos máximos em destaque

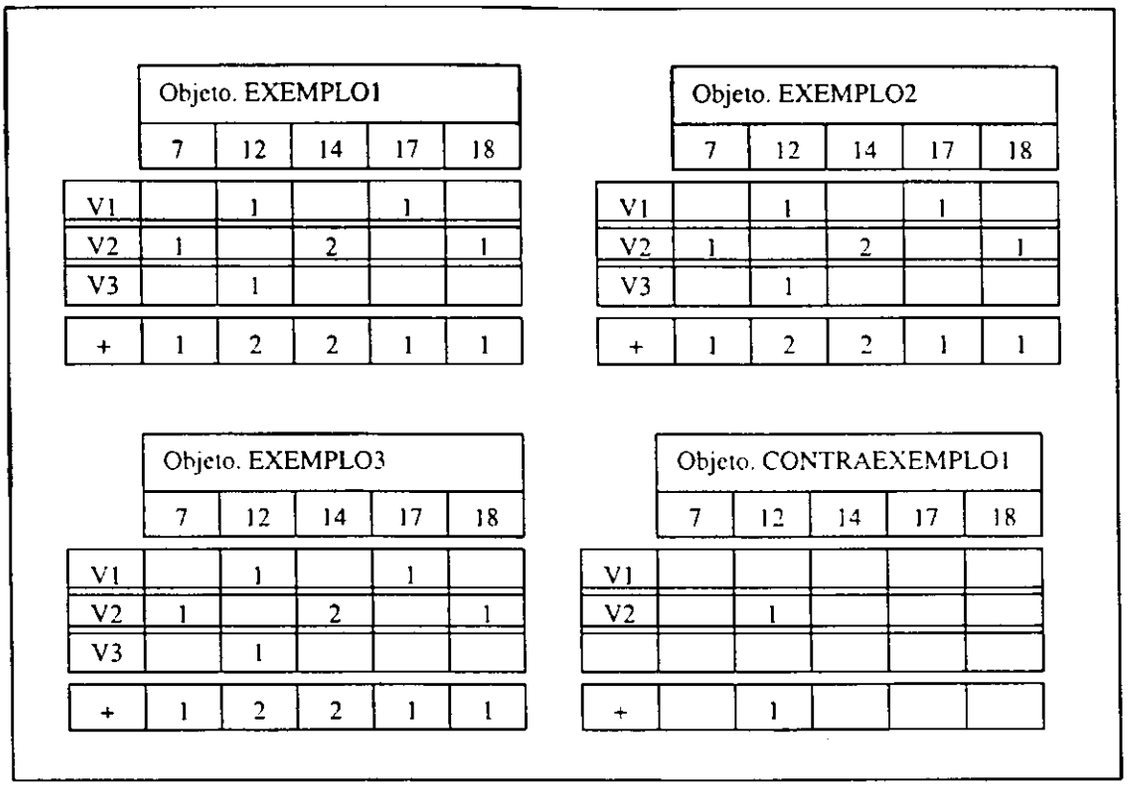


Fig. 4.29 - Avaliação dos FMV Máximos sobre a Amostra

Esse tipo de representação pode ser implementado através de algoritmos proposicionais como o CALM **[QUINQUETON 1986]**. Não é difícil mapear esses resultados para um fórmula da lógica de primeira ordem, por exemplo:

Fato de assimilação: 12

$$\exists\, xy / \mathit{Status}(x,\text{"deitado"}) \& \mathit{Composto\text{-}de}(y,x)$$

Fatos de discriminação: 7 & 12 & 12 & 14 & 14 & 17 & 18

$$\exists x,y,z,k,l,m,n,o,p,q \;/\; \mathit{Status}(y,\text{"deitado"}) \,\&\, \mathit{Composto\text{-}de}(x,y) \,\&\, \mathit{Status}(z,\text{"Em-pé"}) \,\&\, \mathit{Composto\text{-}de}(x,z) \,\&\, \mathit{Status}(k,\text{"Em-pé"}) \,\&\, \mathit{Sobre}(l,k) \,\&\, \mathit{Composto\text{-}de}(x,l) \,\&\, \mathit{Status}(m,\text{"Em-pé"}) \,\&\, \mathit{Esquerda\text{-}de}(m,n) \,\&\, \mathit{Composto\text{-}de}(x,m) \,\&\, \mathit{Status}(o,\text{"Em-pé"}) \,\&\, \mathit{Esquerda\text{-}de}(o,p) \,\&\, \mathit{Sobre}(q,o) \,\&\, \mathit{Composto\text{-}de}(x,o)$$

4.8 Controle Externo

Vamos aproveitar os resultados obtidos para apresentar o motivo de um controle heurístico no S3O. Observe que o conjunto de fatos elementares majoritariamente válidos resultante para os dados de entrada, não inclui o fato <ESTÁVEL,STATUS,EM-PÉ>. Como sabemos, esse fato não foi incluído porque o fato elementar majoritariamente válido <BLOCO,STATUS,EM-PÉ> possui um nível de abstração menor (é menos geral ou mais específico), e o S3O mantém os fatos com menor nível

de abstração possível.

Podemos observar nos grafos que representam os elementos da amostra, que o objeto "ESTÁVEL" é uma classe que abrange os seguintes objetos: BLOCO, QUADRADO, TRIÂNGULO E RETÂNGULO. Ora, sabemos que a generalização no nível de BLOCO já é suficiente para abranger as suas instâncias diretas: BLOCO1, ..., BLOCO7, não sendo necessária a generalização para o próximo nível (ESTÁVEL); sabemos também que os fatos que incluem as figuras QUADRADO1, TRIÂNGULO1 E RETÂNGULO1 foram generalizadas para ESTÁVEL porque suas classes imediatamente superiores não atingiam os limites de validade majoritária. Assim, o S3O não enxerga os BLOCOS do exemplo como ESTÁVEIS, apenas as figuras QUADRADO1, TRIÂNGULO1 E RETÂNGULO1.

Vamos agora considerar que o S3O tenha considerado generalizar os objetos BLOCO para sua classe imediatamente superior (ESTÁVEL). A árvore de fatos gerada para esse exemplo modificado é apresentada na fig. 4.30; O efeito desta mudança afetaria os fatos derivados dos elementares e portanto, os FMV máximos. Como consequência, a construção do objeto generalizante envolveria fatos diferentes, como podemos observar na agregação dos mesmos, como mostra a fig. 4.31. Observe que temos que considerar um número maior de fatos que no exemplo original, e alguns deles são diferentes. Em algumas ocasiões, a alteração de um único fato pode modificar o conjunto inteiro de FMV máximos. Para admitir esse tipo de alteração, o S3O admite a interferência do usuário após encontrar os FEMV, o que permite modificar os fatos majoritariamente válidos. Esta interferência é realizada pela generalização ou especialização de um FEMV, pela inclusão de um outro FEMV modificado, sem alterar o conjunto original ou até mesmo pela remoção de um FEMV que o usuário julgue não ser interessante.

A alteração, inclusão ou remoção de um FEMV é realizada antes do passo da "busca de similaridades". Esta operação é uma simples edição do conjunto de dados gerados na fase de identificação dos FEMV. Para que essa interferência não traga maiores problemas à aprendizagem, o S3O avalia a validade do fato incluído ou modificado antes de realizar a busca de similaridades.

A verificação da validade do fato incluído/editado é realizada pelo S3O da seguinte maneira:

- O S3O verifica se os FEMV originais podem derivar o novo fato (verificação da especialização do novo fato)
- O S3O deriva os fatos possíveis a partir do novo fato e verifica se existe alguma instância possível para o mesmo dentre os FEMV (verificação da generalização do novo fato)
- Finalmente, o S3O verifica se o filtro admite a validação do novo fato.

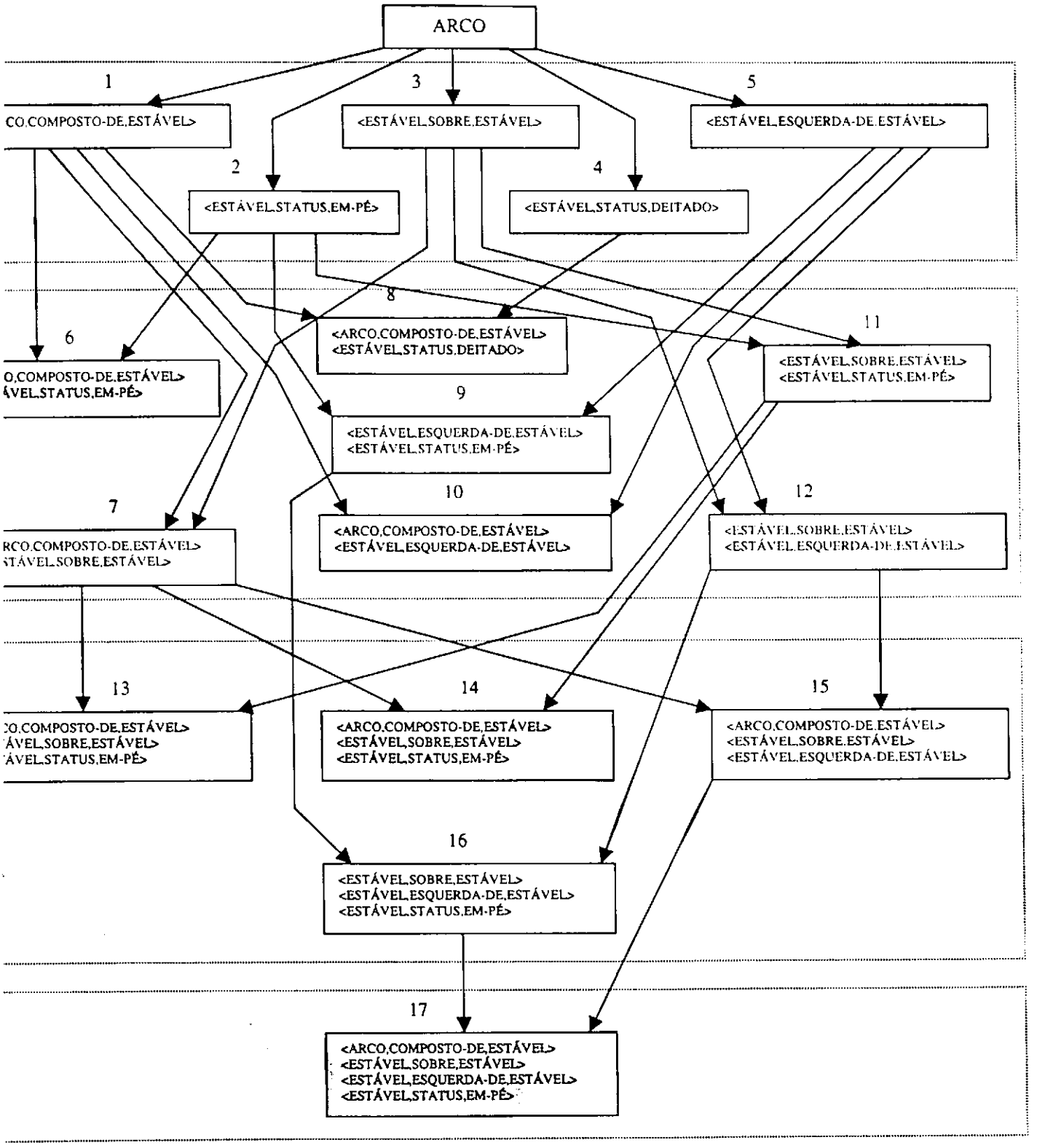


Fig. 4.30 - Árvore de FMVs, com fatos modificados e fatos máximos destacados

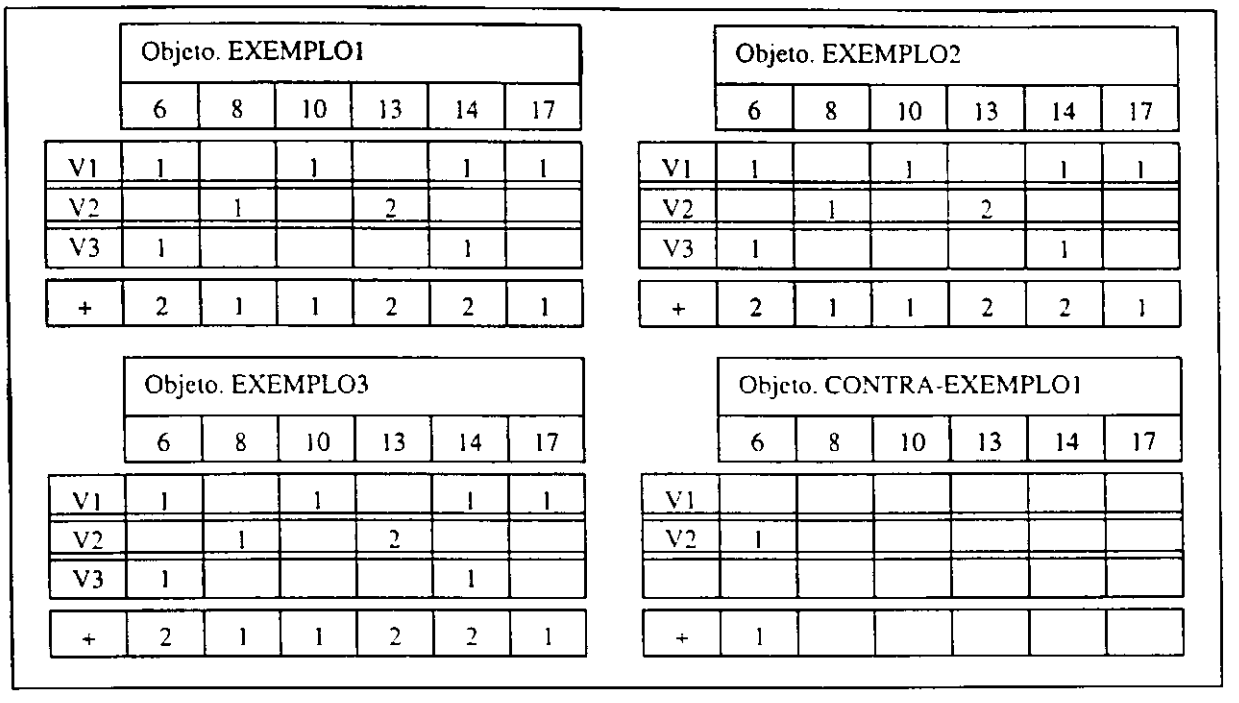


Fig. 4.31 - Avaliação dos FMV Máximos após modificação de um fato

A análise dos efeitos destas alterações poderá ser verificada após a busca de similaridades, onde será possível observar a quantidade dos fatos gerados. Porém, podemos melhor avaliar seu resultado após a geração dos objetos generalizantes de assimilação e discriminação.

A operação de edição ou inclusão do fato modificado é realizada sobre o conjunto dos FEMV já encontrados, os procedimentos de validação do novo fato são realizados da mesma forma que apresentados para os demais. O procedimento de generalização ou especialização de um fato é realizado da seguinte maneira:

- seleciona-se o FEMV que se deseja alterar;
- indica-se o procedimento que se deseja realizar - generalização ou especialização;
- No caso da generalização, o S3O busca todas as classes de nível superior do Objeto e do Atributo do fato, como mostra a fig. 4.32, e, logo após, gera todos os fatos possíveis, sem verificar os limites de validade majoritária ou o filtro. O fato escolhido deve ser indicado pelo usuário, substituindo o fato original.
- No caso da especialização, o S3O busca todas as classes ou instâncias de nível inferior do Objeto e do Atributo do fato, como mostra a fig. 4.32, e, logo após, gera todos os fatos possíveis, verifica os limites de validade majoritária e o filtro. O fato escolhido deve ser indicado pelo usuário, substituindo o fato original.

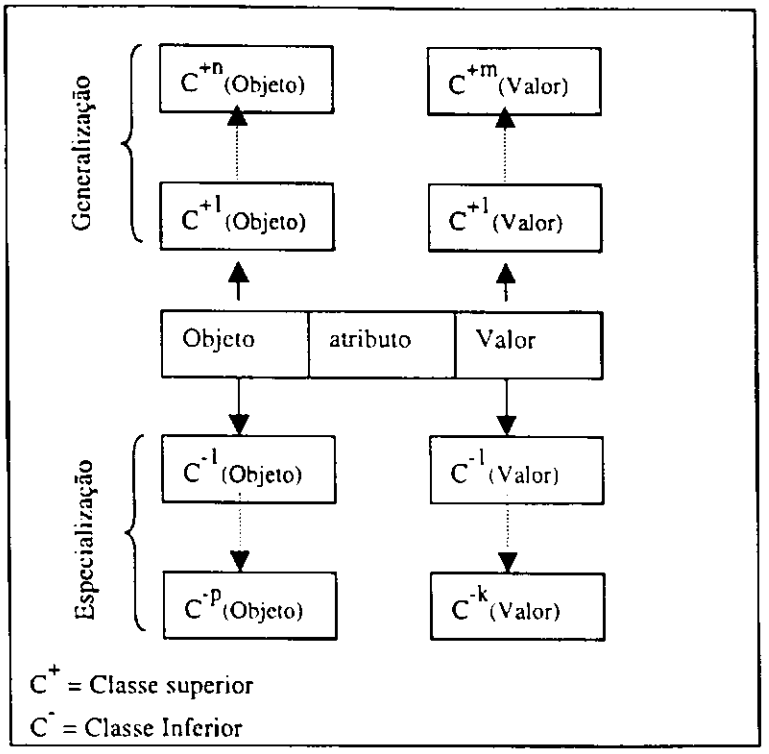


Fig. 4.32 - Generalização ou especialização de um FEMV

Para preservar o conjunto original e incluir um novo fato, o usuário edita um novo fato e o submete ao S3O que avalia o filtro e os limites de validade majoritária e confirmando sua validade, o inclui no conjunto original. De maneira geral, um novo fato incluído só poderá ser diferenciado de um dos FEMV em seu conjunto de valores dos atributos. No caso da exclusão de um FEMV, basta indicar o fato a ser removido e o S3O o retira do conjunto de FEMVs original.

Além das operações já apresentadas de especialização e generalização, esses procedimentos envolvem operações de edição simples; tais operações não serão aqui apresentadas por não serem relevantes para o sistema, mas serão mostradas em sua funcionalidade no capítulo 5.

4.9 Resumo do capítulo

Este capítulo apresentou o S3O, um sistema que realiza busca de similaridades em um conjunto de objetos que representam instâncias positivas e negativas de um determinado conceito. Isto foi feito utilizando-se descrições estruturais denominadas grafos conceituais, que permite a representação de objetos estruturados de forma simples e eficiente.

Vimos que o S3O realiza quatro operações básicas até chegar ao resultado final esperado: a seleção dos fatos elementares dos elementos da amostra, a seleção dentre esses fatos elementares, daqueles que são majoritariamente válidos segundo algumas regras de restrição, a identificação das

trilhas conceituais dos elementos da amostra e finalmente, a busca das similaridades, que consiste na geração a partir dos fatos elementares majoritariamente válidos em estruturas de maior tamanho e busca destas estruturas nas trilhas conceituais identificadas, obtendo o conjunto de fatos majoritariamente validos, que definem em intensão, os elementos da amostra, tanto na interpretação geral, utilizando os fatos majoritariamente válidos de assimilação, quanto na interpretação dos objetos sem ruído, utilizando os fatos majoritariamente válidos de discriminação. Vimos também, que o S3O permite ao usuário a alteração do conjunto de fatos aprendidos, através da manipulação dos níveis de abstração dos fatos elementares majoritários.

Por fim, apresentamos como é feita a construção dos objetos que agregam os fatos majoritariamente válidos (objetos generalizantes de assimilação e de discriminação), que podem ser instanciados por qualquer objeto da amostra segundo a interpretação.

O próximo capítulo apresenta um protótipo do sistema S3O, que em virtude de sua elaboração didática, deve contribuir para o melhor entendimento do que foi aqui apresentado.

Capítulo 5

Protótipo do S3O

5.1 Objetivos do Protótipo

O protótipo do sistema S3O foi construído com as seguintes finalidades:

- Estabelecer uma forma inicial do S3O e avaliar sua funcionalidade;
- Avaliar a qualidade da aprendizagem do sistema;
- Estabelecer um ponto de referência para uma implementação mais aprimorada.

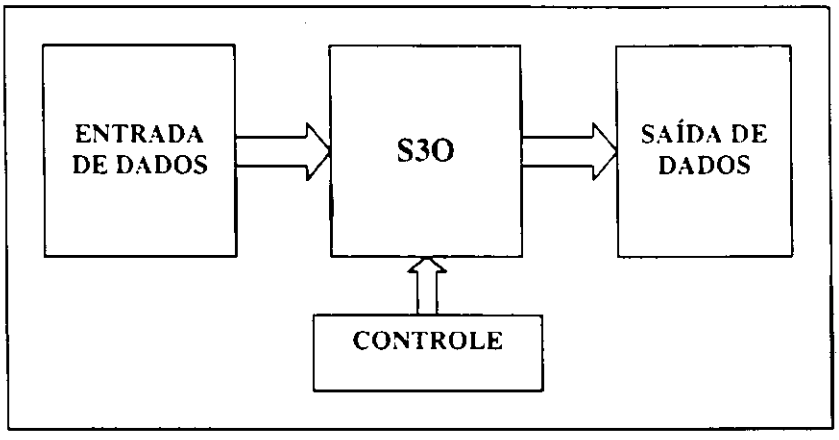


Fig. 5.1 - Módulos do Protótipo S3O

Para construir o protótipo do S3O, foi necessário desenhar uma interface para inicialização dos dados de entrada e exposição dos dados de saída do sistema. Isto porque o S3O em si deve ser chamado como uma rotina do sistema SAID e manipulará os dados existentes no ambiente do mesmo, não possuindo janelas de edição de Classes, Amostra e Classe Principal, mas apenas as janelas de edição do filtro, de restrições de busca e de generalização/especialização. A arquitetura do protótipo está definida como mostra a fig. 5.1.

Cada um dos módulos apresentados na fig. 5.1 é responsável por uma função específica, a saber:

- **Entrada de dados**: este módulo permite a edição dos dados de entrada do S3O: O con-

junto de Exemplos e Contra-exemplos (Amostra), O Filtro de restrições, o nome da Classe Principal e os limites superior e inferior de validade majoritária. No sistema SAID, essas informações deverão estar disponíveis na memória sempre que o S3O for utilizado.

- **S3O**: neste módulo reside a essência do sistema; os algoritmos de identificação dos fatos elementares, de identificação das trilhas conceituais e de busca de similaridades. Esse módulo contém também uma rotina que agrupa as trilhas resultantes dos três primeiros processos acima em um objeto generalizante.
- **Saída de dados**: a saída de dados foi implementada nesse protótipo de maneira bastante didática, mostrando as modificações ocorridas nos dados a cada passo. Assim, esse módulo apresentará na forma de relatórios exibidos no vídeo ou opcionalmente impressos, os seguintes dados: O conjunto dos fatos elementares definidos em extensão e em intensão, as trilhas conceituais, o conjunto dos fatos elementares, as classes de equivalência geradas e finalmente, o conjunto de fatos majoritariamente válidos sobre a amostra.
- **Controle**: nesse módulo é realizada a parte interativa do protótipo, que consiste em controlar o nível de interpretação dos dados, ou seja, percorrer a trilha do grafo estabelecido pela taxonomia de classes. O controle não interativo consiste em estabelecer os valores das variáveis de restrição de busca: Limites positivos e negativos de validade majoritária e o Filtro; variáveis utilizadas pelo módulo de entrada de dados.

O protótipo foi construído a partir das ferramentas de prototipagem do FPW26 **[POWELL 1994]** e do VISUAL FOXPRO 5.0 FOR WINDOWS **[1996]**. O ambiente de desenvolvimento do Foxpro permite o rápido desenvolvimento de aplicações com geração de código otimizado. A estrutura de dados utilizada envolve tabelas indexadas, pilhas e listas encadeadas. Embora sua implementação não possa ser utilizada como uma referência para avaliação em termos de eficiência computacional, instalamos um módulo adicional que fornece um resumo sobre os dados estatísticos dos procedimentos em função do tempo. Esta informação não considera o tempo de execução das rotinas de edição e apresentação de dados, focalizando apenas os procedimentos básicos do S3O.

Dependendo da maneira como for instalado, o protótipo pode ser iniciado com a execução do arquivo S3O.EXE, que necessita dos arquivos de "suporte a runtime". Opcionalmente, podemos executá-lo sob o ambiente FPW26 ou VFP 5.0 através da opção RUN S3O.EXE. Em qualquer caso, recomenda-se que os seguintes arquivos sejam copiados para o diretório onde se localiza o protótipo: KBVIEW.APP (Knowledge Base View), FOXW2600.ESL (Support Library File), FOXPRO.INT (International File), FPWIZ (Wizards) e CATALOG.APP. Alguns desses arquivos já são anexados automaticamente durante a geração do código executável.

5.2 - Janelas do protótipo S3O

5.2.1 - Janela de Abertura

Ao ser iniciado, o protótipo apresenta uma janela de abertura bastante simplificada, consistindo basicamente de 9 botões; esta janela é mostrada na fig. 5.2; dois conjuntos de botões que ativam procedimentos funcionalmente distintos podem ser facilmente identificados: **Definições dos dados** e **Operações**. Nesses conjuntos estão disponíveis os seguintes botões de ativação:

- **Conjunto de botões para definições de dados:**

  Classes: acesso à janela de edição de Classes (5.2.2)

  Instâncias: acesso à janela de edição de Instâncias (5.2.3)

  Classe Principal: acesso à janela de definição da Classe Principal (5.2.4)

  Amostra: acesso à janela de definição da Amostra (5.2.5)

  Filtro: acesso à janela de definição do Filtro (5.2.6)

  Restrições de Busca: acesso à janela de definição dos dados de controle da busca de similaridades (5.2.7)

- **Conjunto de botões para operações:**

  Busca de Similaridades: esta opção ativa a janela de abdução de dados, que apresenta as opções de ativação dos procedimentos que manipulam os dados editados a partir das janelas de edições de dados (classe principal, classe, instância, etc.). Esta janela, bem como as que por ela são ativadas, estão descritas em 5.2.8.

  Objeto Generalizante: esta opção inicia o procedimento de construção dos objetos generalizantes, que são gerados a partir da abdução de fatos. Esse procedimento não pertence ao sistema S3O, mas foi incluído para indicar o tratamento dos dados após a execução do S3O .

Cada uma das janelas chamadas a partir do menu principal é apresentada nas seções seguintes (5.22 a 5.28); além destas opções, também encontra-se disponível, o botão para encerramento da operação com o S3O; o botão "sair" .

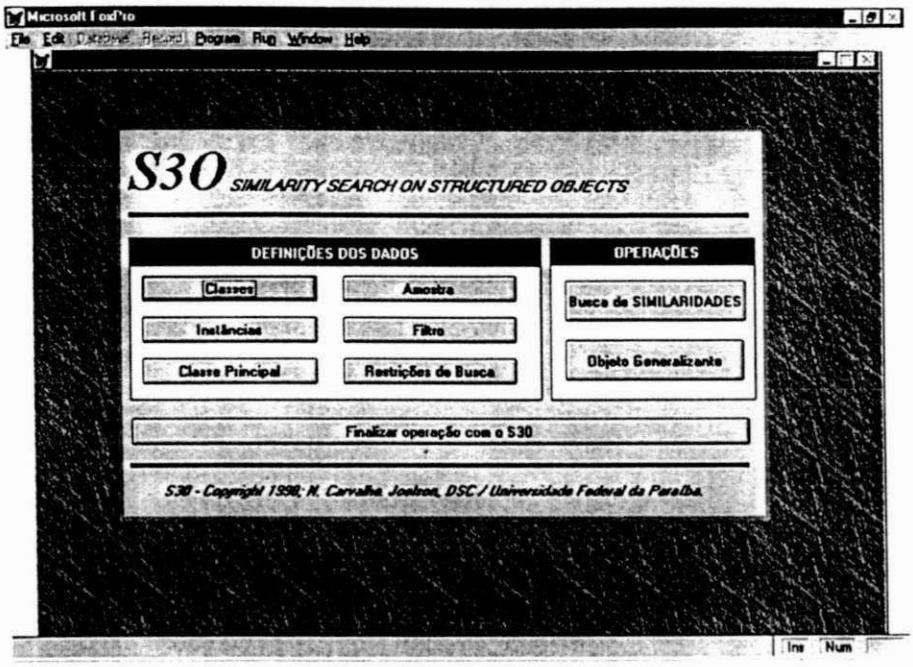


Fig. 5.2 - Janela de Abertura do S3O

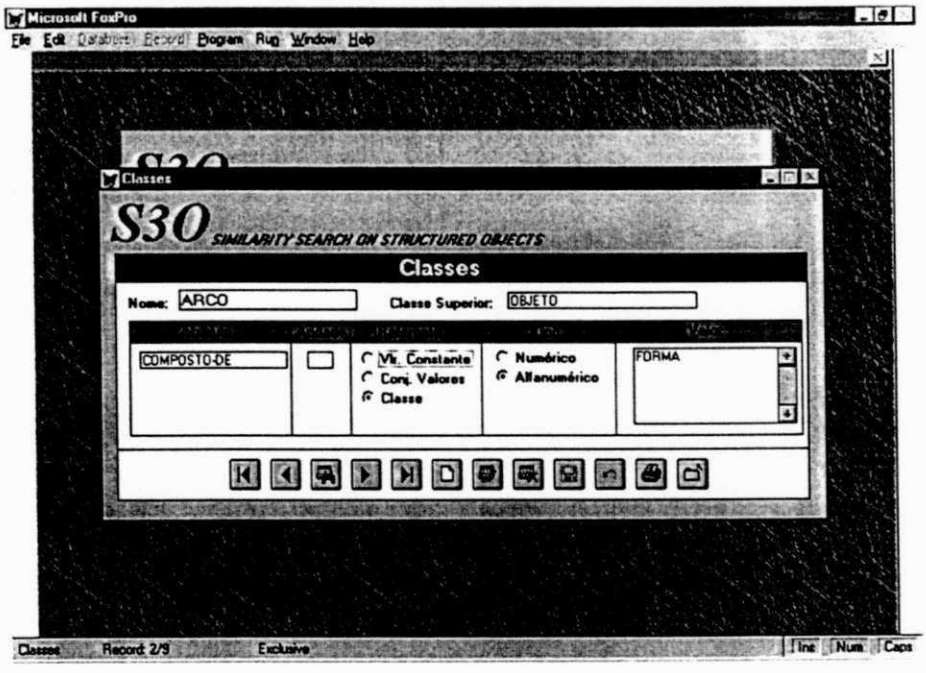


Fig. 5.3 - Janela de Definição das Classes

5.2.2 - Janela de Edição das Classes

A janela de edição das Classes apresentada na Fig. 5.3 permite a edição dos seguintes dados relativos às classes:

- **Nome**: indica o nome da classe;
- **Classe superior**: indica a classe de nível hierárquico (taxonomia) imediatamente superior à classe sendo editada. No caso da classe ser definida como "Objeto Principal", a informação da classe superior deve obrigatoriamente ser "OBJETO". Esta informação permite estabelecer uma hierarquia de classes;
- **Atributo**: para cada classe, um conjunto de atributos pode ser disponibilizado; Esse campo indica o nome de um dos atributos pertencente à classe indicada por "Nome". Todas as demais informações desta janela são pertinentes ao atributo;
- **Aridade**: indica quantas unidades do valor (no caso de o valor ser composto por um conjunto de valores) o atributo requer como referendo. Por exemplo, para o atributo "língua-estrangeira-conhecida", o valor poderia ser o conjunto (inglês, francês, alemão, espanhol), e a aridade poderia variar de 1 a 4;
- **Tipo**: esta informação indica o tipo do valor (ou valores) do referente e é utilizada para simplificar a realização de operações aritméticas e de caracteres.
- **Referente**: indica como está disponibilizada a informação sobre o valor do referente; se é um valor constante, se é formado por um conjunto de valores, ou ainda se é uma classe de valores.
- **Valor**: esta variável indica o valor do referente, e deve seguir a indicação da variável "Referente". No caso desse valor ser uma constante ou uma classe de valores, basta digitar o valor numérico ou alfabético (sem aspas). No caso de o referente ser um conjunto de caracteres, os valores devem ser separados por vírgula, sem espaços entre eles, não sendo necessário vírgula ou ponto após o último valor da série.

Podemos identificar na Fig. 5.4 botões que permitem a manipulação dos dados; esses botões estão presentes em quase todas as janelas de edição do S3O.

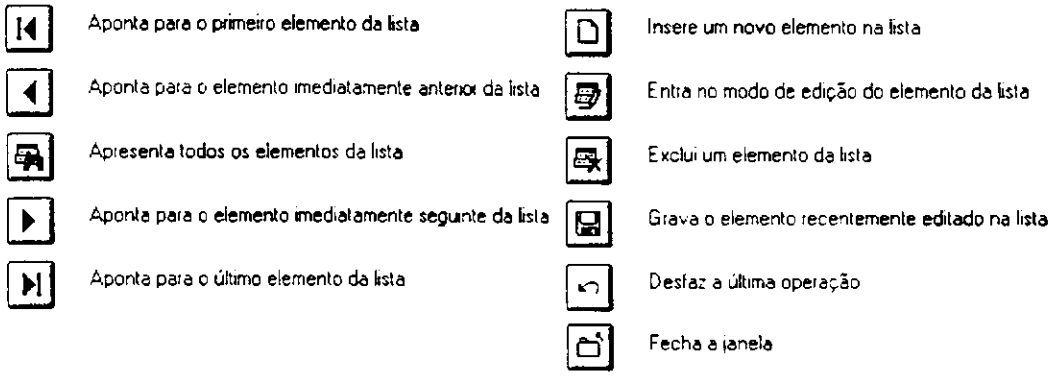


Fig. 5.4 - Botões de edição

5.2.3 - Janela de Edição das Instâncias

A fig. 5.5 apresenta a janela de edição dos dados das instâncias que devem ser informados como segue:

- **Nome**: indica o nome da instância;
- **Classe superior:** indica a classe a qual pertence a instância sendo editada, cuja definição já deve ter sido feita na janela apresentada na seção anterior;
- **Atributo**: para cada instância, um conjunto de atributos pode ser disponibilizado. Esse campo indica o nome de um dos atributos pertencente à instância indicada por "Nome". Todas as demais informações desta janela, são pertinentes ao atributo, e são exatamente como descritos na janela de edição de classes (seção 5.2.2);
- **Aridade**: indica quantas unidades do valor (no caso de o valor ser composto por um conjunto de valores) o atributo requer como operandos.
- **Referente**: indica como está disponibilizada a informação sobre o valor do referente; se é um valor constante, se é formado por um conjunto de valores, ou ainda se é uma classe de valores; nesse último caso, um dos atributos da instância seria uma classe inteira de valores;
- **Tipo**: esta informação indica o tipo do valor, com as opções *Numérico* ou *Alfabético*;
- **Valor**: esse campo deve conter o valor do referente.

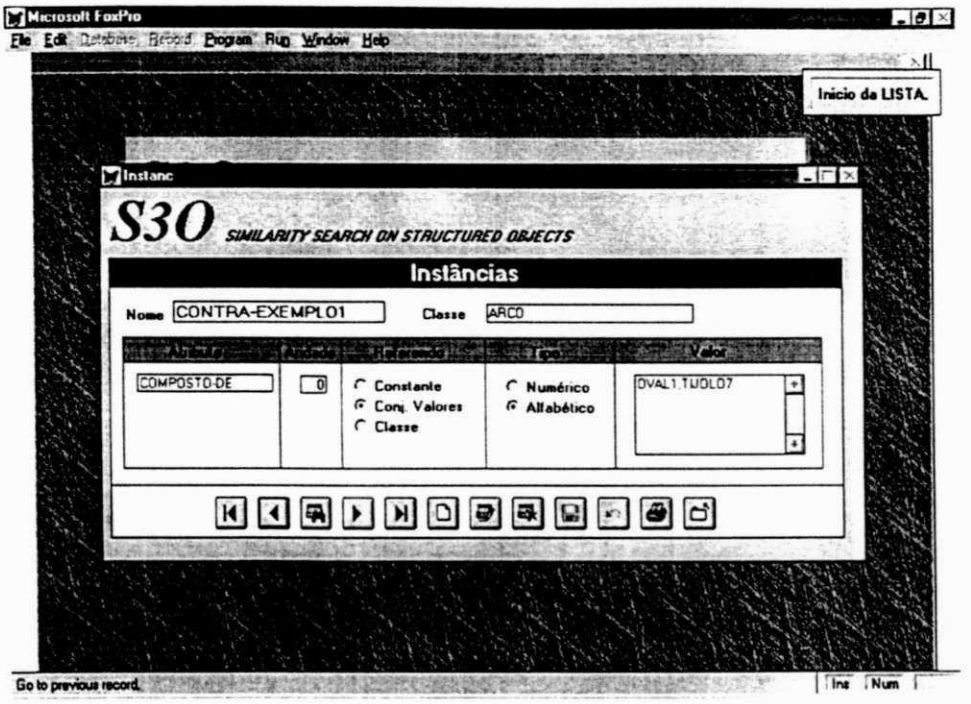


Fig. 5.5 - Janela de Definição das Instâncias

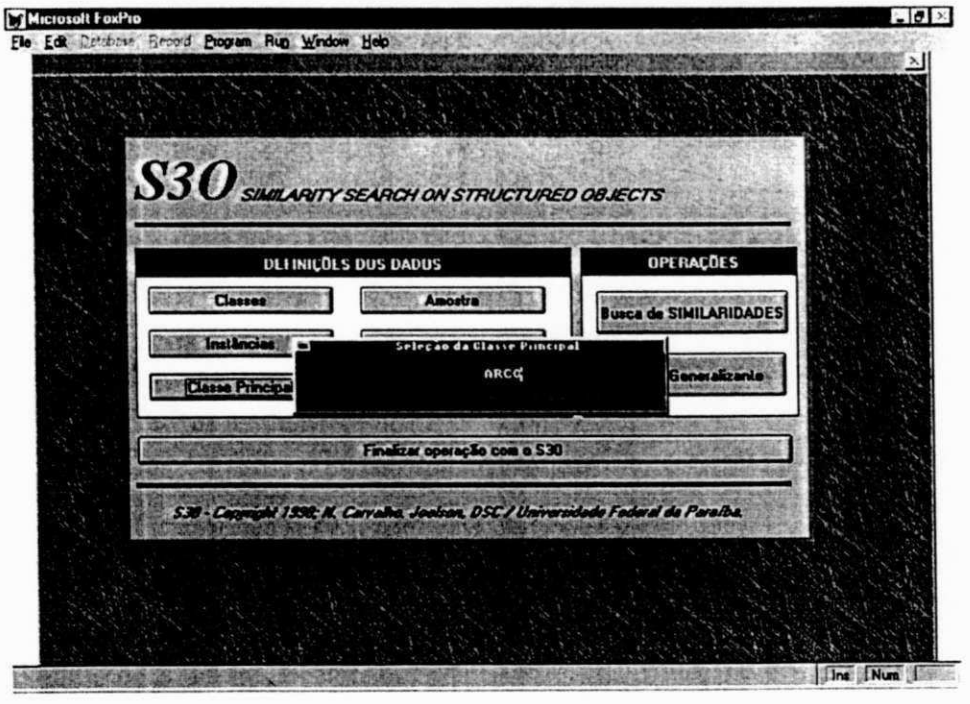


Fig. 5.6 - janela de definição da classe principal

5.2.4 - Janela de Definição da Classe Principal

Esta janela (Fig. 5.6) permite que seja indicada a classe principal sobre a qual será realizada a

aprendizagem (em outras palavras, indica o nome do conceito a ser aprendido).

5.2.5 - Janela de Definição da Amostra

Esta janela permite indicar, dentre os elementos já instanciados, aqueles que serão considerados Exemplos e aqueles que serão Contra-Exemplos de objetos da Classe Principal (que já deverá estar previamente definida). Inicialmente, o sistema pesquisa seus dados procurando por aqueles cujo atributo *tipo-de* esteja ligado à Classe Principal. Os objetos identificadores desses elementos são então selecionados e apresentados numa lista, como mostra a Fig. 5.7, para que o usuário defina, para cada um dos elementos, se deve ser considerado "Exemplo" ou "Contra-exemplo", marcando um dos botões (radio buttons) que aparecem ao lado. Note que o sistema apresenta apenas o primeiro elemento da lista, os demais devem ser indicados em seguida, utilizando para isso, a seta que aponta para o elemento imediatamente seguinte da lista.

5.2.6 - Janela de Definição do Filtro

A janela de definição do filtro, apresentada na fig. 5.8, permite que o usuário do S3O insira restrições para a análise de validade dos fatos elementos da amostra, através da especificação de valores possíveis para um determinado atributo; por exemplo, considerando o fato <Carro, Cor, {1, vermelho, verde, amarelo}> um usuário pode especificar que não deseja considerar os fatos desse tipo que tenham o valor "amarelo". Basta então especificar no filtro, os dados do fato (classe principal, classe, atributo, a função de comparação a ser usada (no caso, =) e a cor amarelo.

A característica do filtro é restritiva, por esse motivo, só deve ser utilizado para especificar os valores dos atributos que não devem ser considerados. As informações inseridas em cada entrada são cumulativas, isto é, existe uma função "AND" implícita entre as mesmas; assim, se indicarmos em uma entrada duas restrições, apenas os fatos com essas duas restrições serão filtrados; os fatos que possuem apenas uma das restrições não serão filtrados. Esta janela permite a edição dos seguintes dados:

- **Filtrar para os fatos da Classe Principal**: Indica a classe principal à qual pertence a classe do objeto cujos atributos serão restringidos; esta informação é obrigatória.
- **Filtrar a Classe de Objetos**: especifica uma classe a ser filtrada.
- **Filtrar Objetos definidos pela função**: especifica uma função para filtrar um conjunto específico de valores.

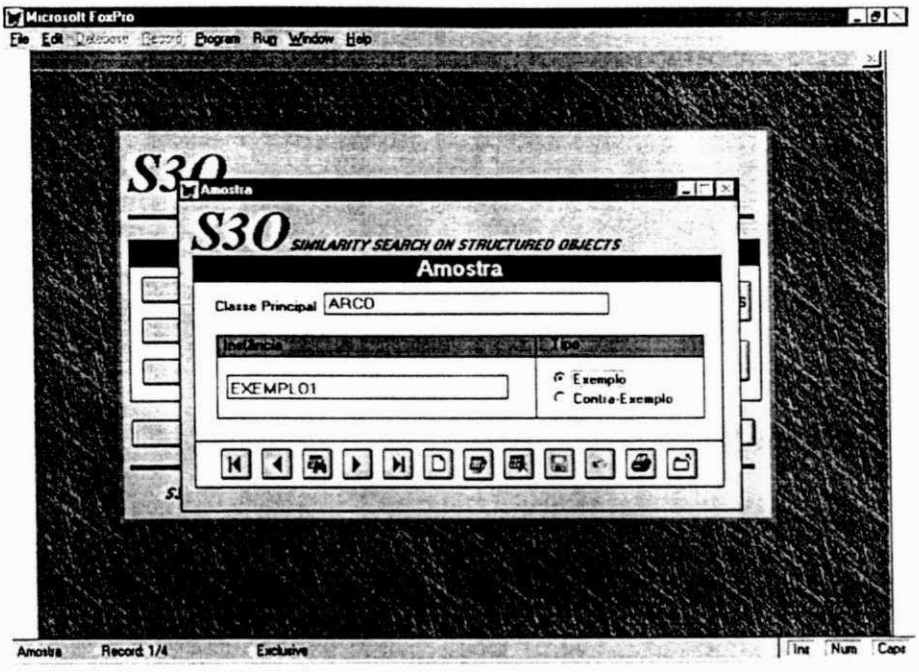


Fig. 5.7 - janela de definição da Amostra

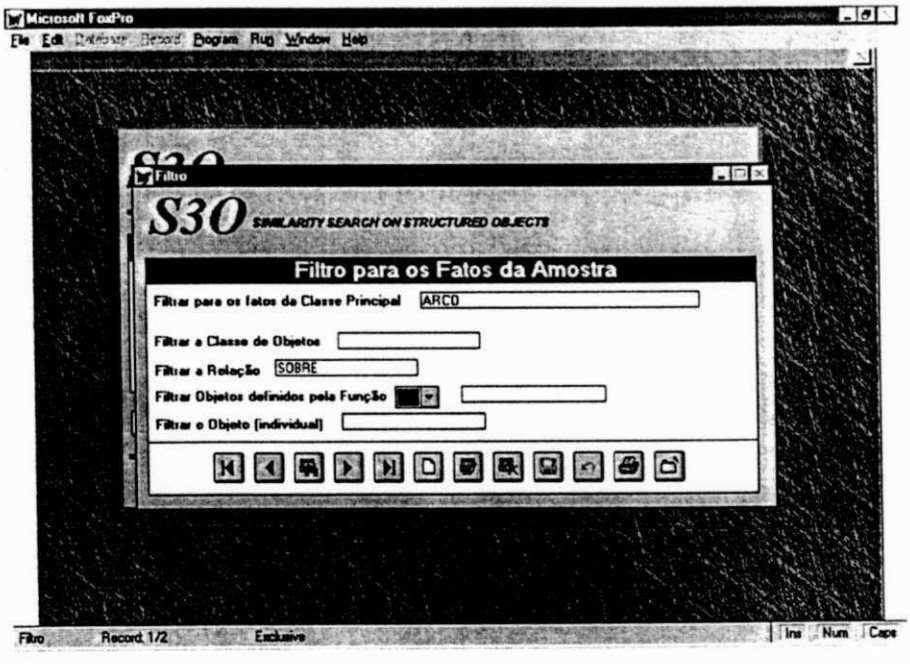


Fig. 5.8 - Janela de Definição do Filtro

- **Filtrar a Relação**: especifica o nome de um atributo a ser filtrado.
- **Filtrar o Objeto individual**: indica o nome de uma Instância a ser filtrada.

Se especificarmos, por exemplo, para a classe principal "ARCO", Filtrar a classe "TIJOLO" e Filtrar a Relação "SOBRE", todos os fatos <O,A,V> definidos por Objeto = "Tijolo" e Atributo = "SOBRE" serão filtrados, não interessando o valor V.

5.2.7 - Janela de Definição dos Limites de Validade Majoritária e Tipo de Interpretação

Esta janela permite definir os Limites Positivo e Negativo de Validade Majoritária, isto é, o percentual de Exemplos e Contra-Exemplos que deverão ser obrigatoriamente considerados na Interpretação.

As informações necessárias são, como mostra a fig. 5.9, as seguintes:

- **Classe Principal**: a Classe Principal, como descrita em 5.2.5;
- **Limite Positivo de Validade Majoritária**: o percentual de exemplos sobre a Amostra a considerar;
- **Limite Negativo de Validade Majoritária**: o percentual de Contra-Exemplos sobre a Amostra a considerar;
- **Comprimento Máximo dos Fatos**: é o comprimento máximo de um fato construído pela concatenação dos FEMV;
- **Interpretação**: tipo de interpretação que se deseja, dentre os seguintes:

  Discriminação: interpreta os fatos discriminando as Instâncias, permitindo a distinção entre os Exemplos e os Contra-exemplos;

  Assimilação: caracteriza todo o conjunto de objetos presentes na Amostra;

  Ambos: processa os dois tipos de Interpretação, armazenando o resultado em conjuntos distintos.

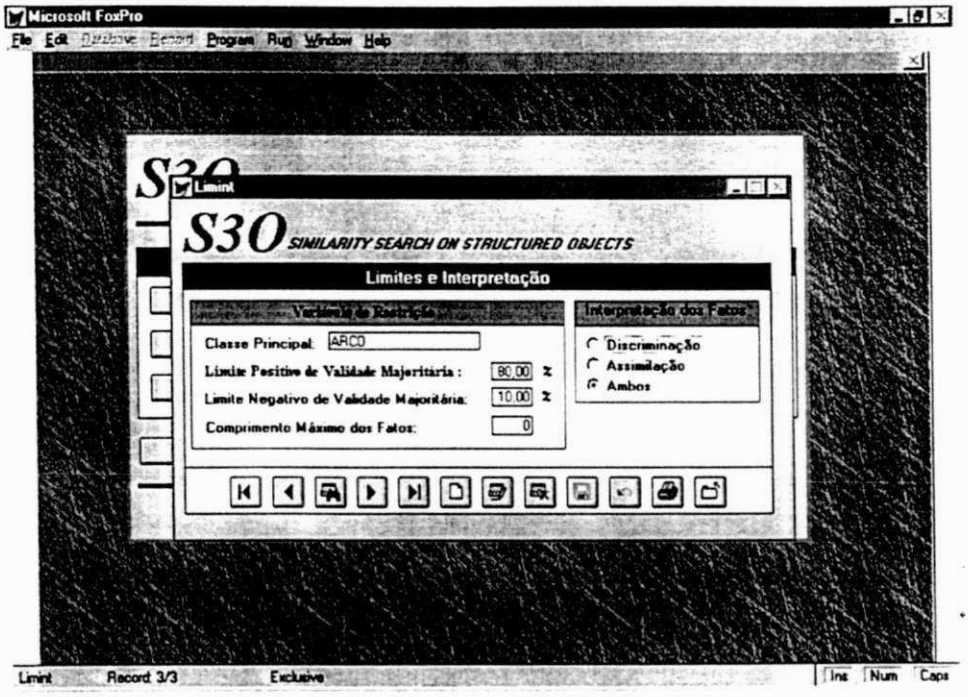


Fig. 5.9 - Janela de definição das variáveis de restrições

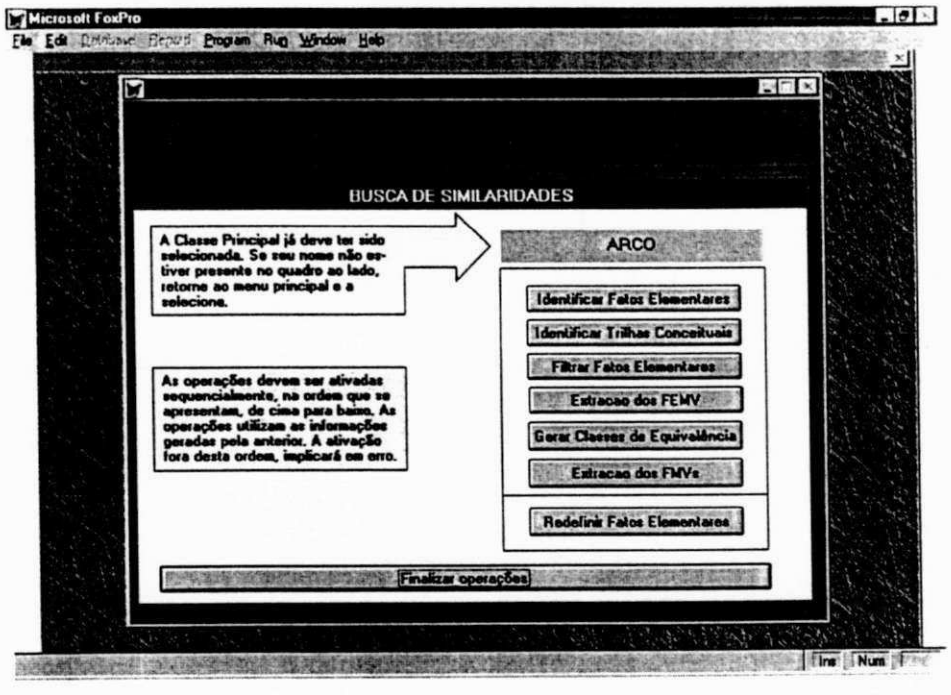


Fig. 5.10 - Janela de Busca de Similaridades

5.2.8 - Janela de Busca de Similaridades

A fig. 5.10 apresenta a janela de Busca de Similaridades. Ela contém o menu de chamada dos procedimentos que manipulam os dados inseridos pelas janelas anteriores: poderíamos dizer que se trata da principal janela do protótipo. Nela, além da indicação da classe principal especificada como mostrado em 5.2.4, estão disponíveis os botões que acionam as seguintes funções ou procedimentos.

5.2.8.1 – Identificação dos Fatos Elementares

Esse botão aciona o procedimento que lê os dados das instâncias e classes, associadas à classe principal selecionada. Esse procedimento, bem como os subseqüentes, apenas funcionarão se a classe principal já estiver especificada, com seu nome indicado no quadro logo acima desse botão. O sistema identificará os fatos elementares para os elementos da amostra vinculados à classe principal e os apresentará no formato de um relatório no monitor de vídeo. Para o exemplo descrito no capítulo anterior, a janela apresentada nas figs. 5.11 e 5.12 apresenta a relação de fatos elementares identificados em extensão e intensão. As apresentações dos dados podem ser roladas na tela (scroll) sempre que o número desses fatos seja superior ao número de linhas do relatório.

5.2.8.2 – Identificação das Trilhas Conceituais

Esse procedimento identifica as trilhas conceituais dos objetos da amostra e os apresenta na tela, como mostra a Fig. 5.13. A relação inclui a informação sobre o tamanho das trilhas, um índice que liga cada uma das trilhas ao fato original (elemento do conjunto Amostra) .

5.2.8.3 – Filtragem dos Fatos Elementares

Esse botão acionar a filtragem dos fatos elementares, segundo o filtro previamente estabelecido e apresenta a relação de definições de filtro utilizadas, como mostra a Fig. 5.14 . Em seguida, a relação dos fatos já filtrados é mostrada.

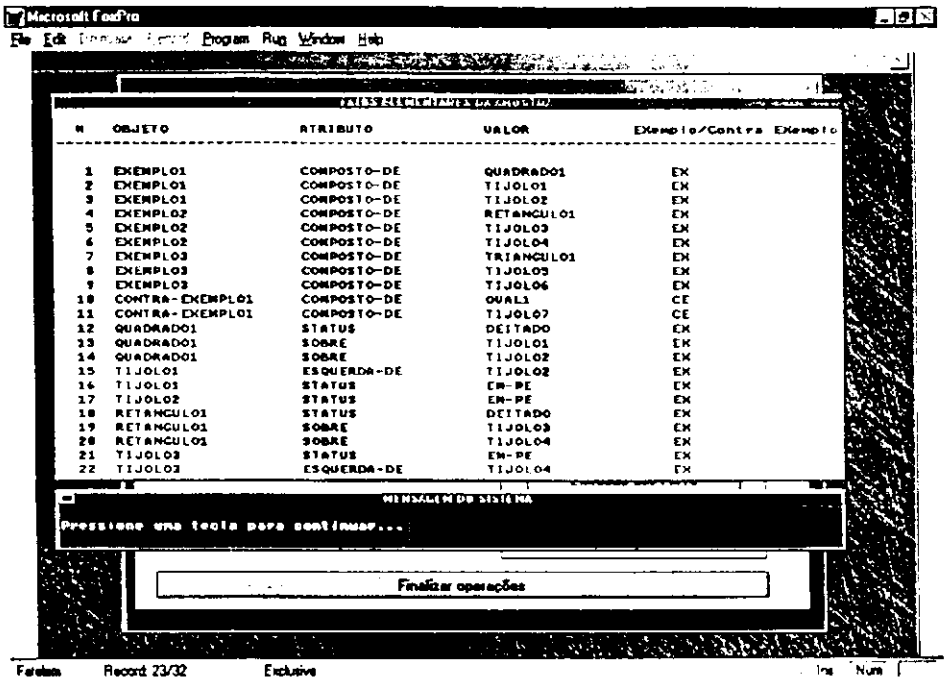


Fig. 5.11 – Relação dos Fatos Elementares da Amostra (extensão)

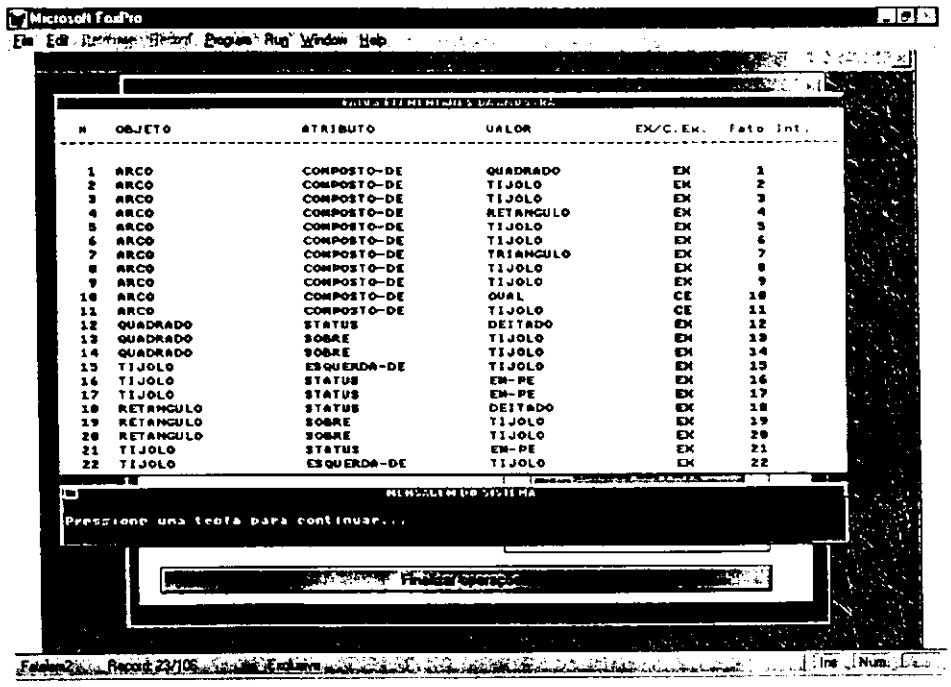


Fig. 5.12 – Relação dos Fatos Elementares da Amostra (intensão)

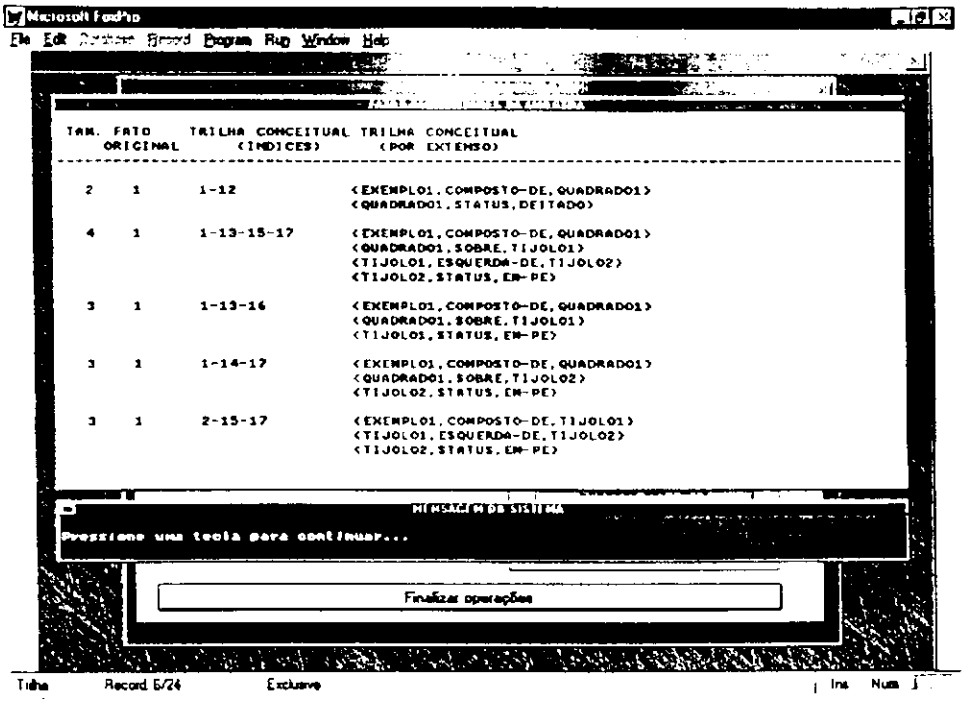


Fig. 5.13 – Relação das Trilhas Conceituais

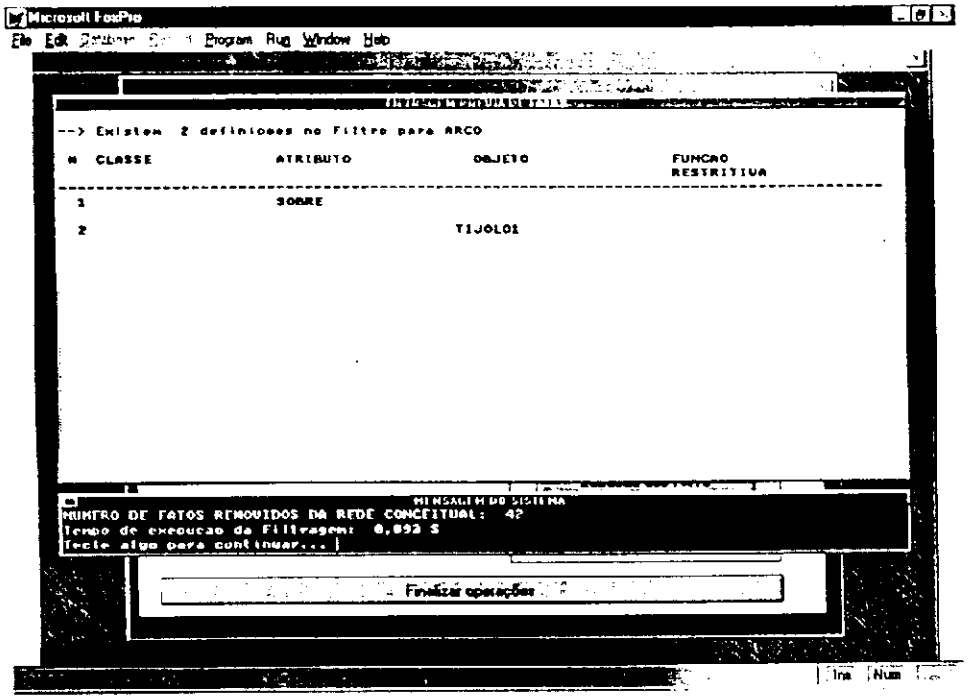


Fig. 5.14 – Definições do Filtro

5.2.8.4 - Extração dos FEMV

Esse botão inicia o procedimento da identificação dos Fatos Elementares Majoritariamente válidos, isto é, o conjunto de Fatos Elementares apresentado em 5.2.8.1, excetuados os fatos filtrados pelo procedimento anterior e pelos limites de validade majoritária. A relação sintética dos FEMVs é apresentada como mostra a Fig. 5.15.

5.2.8.5 – Geração das Classes de Equivalência

Esse botão inicia a geração das classes de equivalência. As classes de equivalência são geradas a partir dos FEMVs por um processo de casamento de padrões (Algoritmo de Boyer-Moore modificado). A relação de classes de equivalência é apresentada na tela conforme mostra a Fig. 5.16.

5.2.8.6 - Extração dos FMVs

Cada uma das classes de equivalência geradas no item anterior deverá respeitar os limites de validade majoritária. Na verdade, a extração dos FMVs é realizada durante a geração das classes de equivalência, pois cada classe gerada é imediatamente submetida às restrições de validade majoritárias e portanto, no final, teremos obtido os Fatos Majoritariamente Válidos sobre a Amostra (FMVs). Esse botão apenas apresenta na tela a relação dos mesmos, como mostra a Fig. 5.17.

5.2.8.7 – Redefinição de Fatos Elementares

Esse botão inicia o procedimento de alteração dos FEMVs pelos procedimentos de Generalização e Especialização. A janela mostrada na Fig. 5.18 permite realizar a edição dos FEMVs (um fato por vez). As operações são realizadas individualmente para os Objetos e para os Valores, clicando sobre o botão específico da operação (Generalização ou Especialização). Após esse procedimento, os fatos devem ser novamente processados desde a Filtragem (5.2.8.3) para que possam ser novamente validados.

Para retornar à janela de abertura, basta clicar sobre o botão **Volta ao menu principal.**

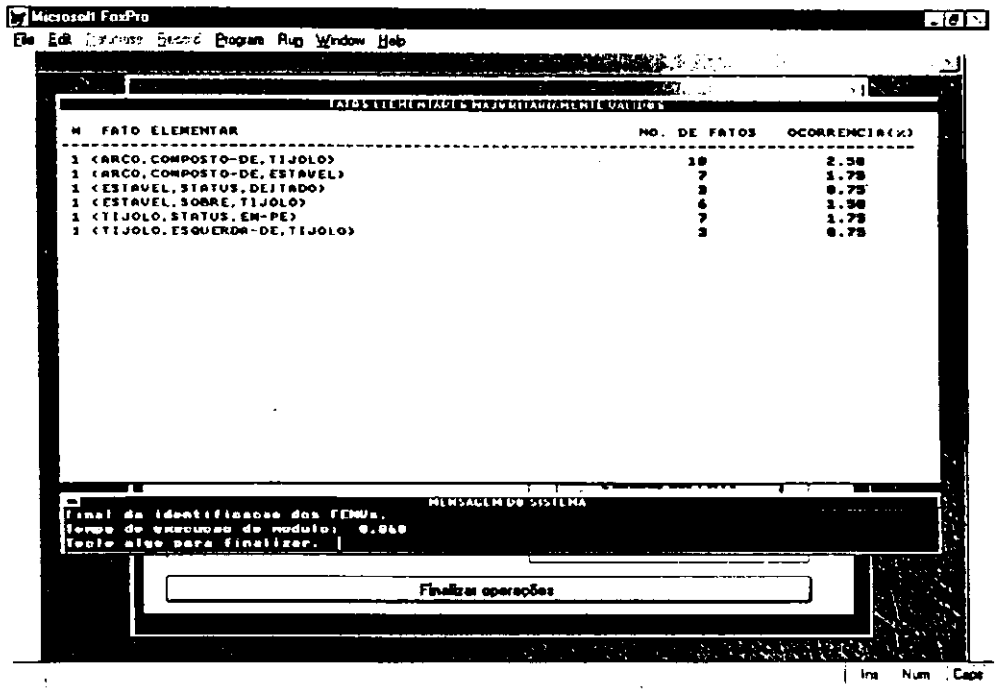


Fig. 5.15 – Relação dos fatos elementares majoritariamente válidos

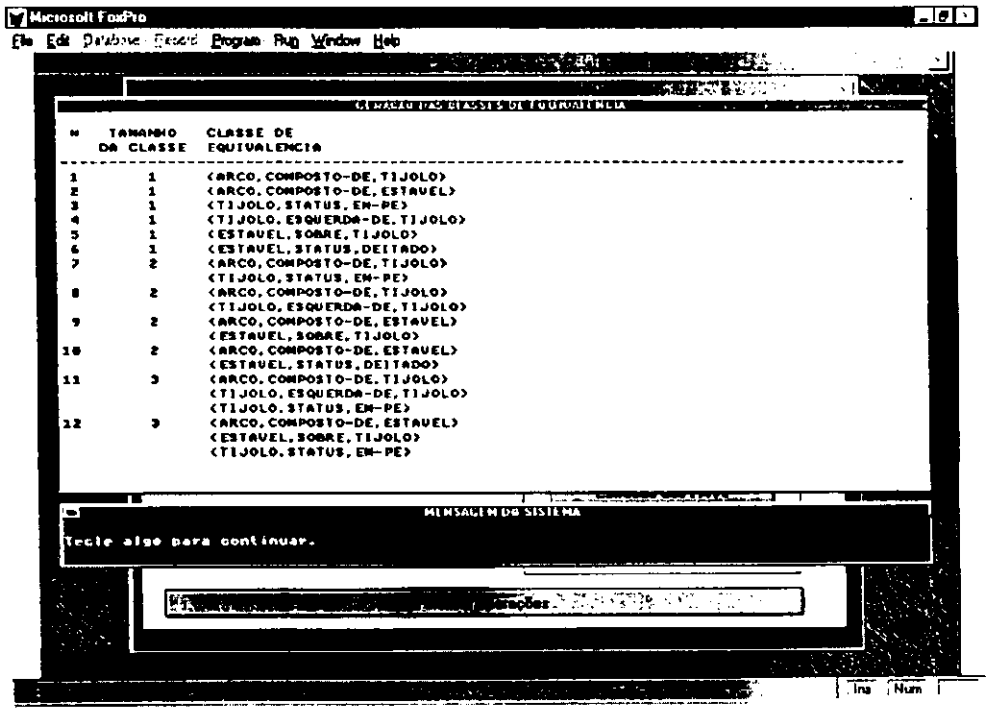


Fig. 5.16 – Relação das classes de equivalência

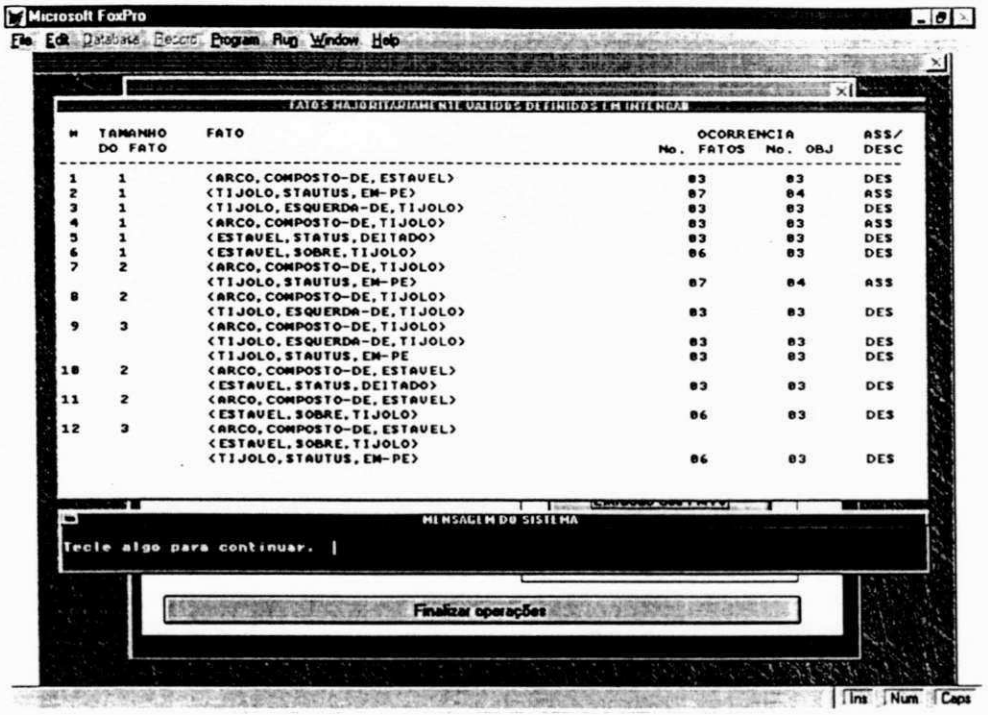


Fig. 5.17 – Relação dos FMVs

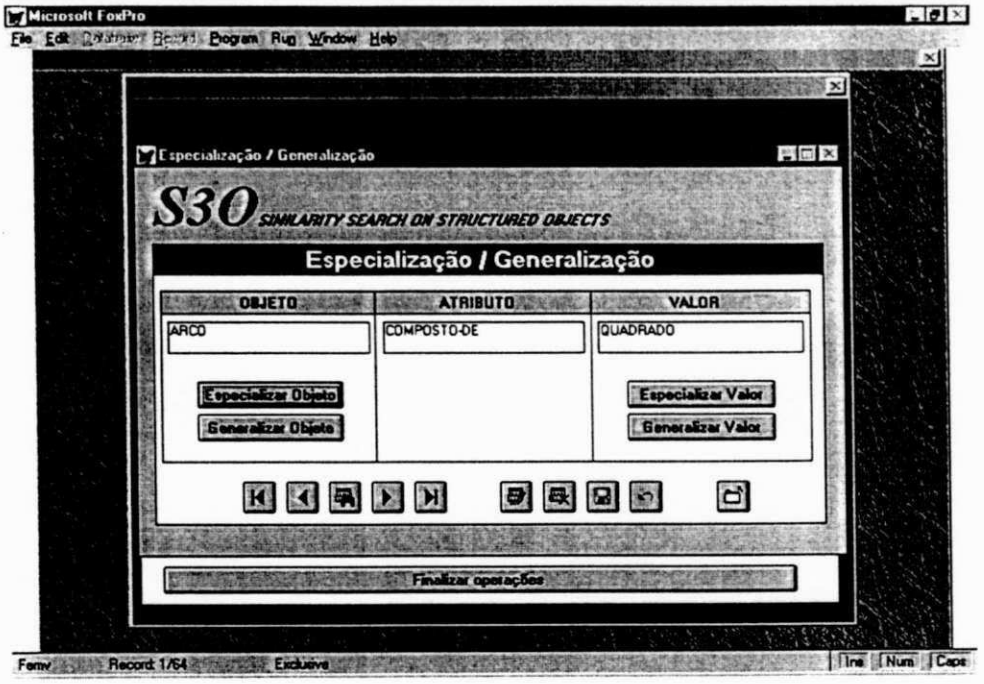


Fig. 5.18 – Janela de redefinição dos FEMVs

5.3 - Resumo do capítulo

Apresentamos nesse capítulo o protótipo S3O, um sistema capaz de construir hipóteses sobre um conceito a partir de objetos que são exemplos e contra-exemplos de objetos do mesmo.

O protótipo é constituído de uma ambiente interativo, que possibilita a edição dos elementos da amostra (constituídos dos conjuntos *classes* e *instâncias*), dos dados de restrição de busca (filtros e limites de validade majoritária) e da definição da *classe principal*. A interface com o usuário é feita através de janelas e além das operações de edição dos dados, o protótipo permite que os vários passos do algoritmo de aprendizagem sejam ativados de maneira seqüencial com pausas entre os mesmos, permitindo que os resultados intermediários sejam apresentados. Finalmente, existe a possibilidade de alteração dos fatos elementares a considerar, que permitem modificar o resultado da aprendizagem.

O objetivo desse protótipo é fundamentalmente didático e permite ao usuário avaliar os dados envolvidos na aprendizagem sob vários aspectos.

Apresentaremos a seguir, a conclusão deste trabalho e nossas perspectivas para trabalhos futuros.

Capítulo 6

Conclusões e Perspectivas

6.1 Conclusões

Nos últimos anos, a aprendizagem de máquina evoluiu de modelos de laboratório para aplicações de grande valor comercial. Os algoritmos de aprendizagem baseada em exemplos envolvidos nesse contexto tem aprendido a realizar inúmeras tarefas e as pesquisas agora focalizam sobre questões envolvendo principalmente o número de elementos para treinamento, o número de hipóteses consideradas em relação à hipótese aprendida, a consistência e a persistência desta hipótese aprendida em função do tempo.

A grande maioria dos sistemas de aprendizagem baseados em similaridades empregados na aquisição automática de conhecimento, independente da sua aplicação, utilizam um conjunto finito de similaridades encontradas com certa regularidade em um conjunto de objetos formado por elementos característicos do conceito a ser aprendido.

Apresentamos neste trabalho o S3O, um sistema de aprendizagem a partir de objetos estruturados que gera um espaço de busca de maneira Top-down e o explora de maneira Bottom-up, e que segue a abordagem geral dos sistemas citados no parágrafo anterior, mas contém algumas características próprias, como um controle interativo da aprendizagem por parte do usuário, a utilização de uma estrutura bastante simples e consistente para suportar os fatos e o tratamento do ruído presente na amostra.

Como foi baseado no método INNE, vale a pena ressaltar as diferenças e vantagens obtidas:

- A representação utilizada nos insere no paradigma da programação orientação a objetos, trazendo todos os benefícios desta abordagem.

- A estrutura adotada para a tupla <Objeto-Atributo-Valor> permite efetivamente melhorar a resistência ao ruído, pois as restrições contextuais são mais fácil e minuciosamente analisadas; além do mais, cada fato é gerado uma única vez e sua estrutura permanece inalterada durante todo o processo, pois os dados derivados não alteram a estrutura do seu gerador.

- Em virtude do estabelecimento inicial da tupla Objeto-Atributo-Valor, o espaço de busca gerado descarta relacionamentos que não se verificam nos fatos do conjunto amostra; isto é, não existe, como no INNE, fatos do tipo "Objeto,Atributo" ou "Atributo-Objeto". Desta maneira, elimina-se grande parte dos possíveis "casamentos" de fatos que seriam mais tarde descartados. Uma outra conseqüência é que os conjuntos de classes de equivalência gerados no S3O são menores.

- A interação com o usuário permite que a aprendizagem seja orientada de maneira a moldar-se a situações tão próximas às expectativas do mesmo quanto possível, através do uso de filtros restritivos e da mudança empírica dos limites de validade majoritários.

Como os sistemas de aprendizagem desta classe, o S3O contém suas limitações, sendo as mais importantes, primeiro, a capacidade de representação dos elementos do conjunto amostra que podem não ser suficientes para caracterizar totalmente um conceito (por exemplo, a amostra pode conter um número muito reduzido de elementos ou um grande número de elementos idênticos) e segundo, a função de avaliação desses elementos (no nosso caso, o estabelecimento dos limites de validade majoritária, cujo ajuste é inicialmente incerto e progride empiricamente). O controle interativo representa um outro problema importante, pois existe o perigo da geração de um conjunto viciado de fatos generalizantes, caso em que um conceito gerado a partir de tais fatos teria uma forte componente pessoal do usuário, e o sistema aprenderia o conceito segundo um "ponto de vista" do mesmo e não segundo a exata expressão da realidade. Por fim, a efetiva avaliação da aprendizagem deve ser avaliada pelo usuário. Uma vez que o sistema gere um conjunto de fatos majoritariamente válidos, ele terá aprendido algo a partir dos exemplos apresentados, mas isto não significa que o conteúdo de sua aprendizagem está interessante para o usuário.

6.2 Perspectivas

Embora sua aplicação primária esteja nesse trabalho orientada para o Agente Racional SAID, o desenvolvimento do S3O nos permitiu vislumbrar um horizonte de várias outras aplicações interessantes que variam desde a aprendizagem de expressões homólogas para uso em tradutores (dois textos semelhantes, expressando o mesmo conteúdo, é uma fonte para encontrar expressões sintaticamente diferentes com o mesmo significado) em linguagem natural , passando pela aprendizagem de conceitos matemáticos básicos da geometria analítica para tutoriais inteligentes, até o uso em um ambiente de sistema CAD/CAM altamente adaptável, que possa assistir o projetista através dos diferentes passos do desenvolvimento do seu projeto. Uma outra interessante possibilidade é a derivação, a partir do S3O, de um sistema de aprendizagem por descrição funcional, que pode ser

obtido com algumas poucas modificações. Alguns desses trabalhos já estão sendo iniciados.

Nossos próximos trabalhos focalizarão as limitações do S3O; inicialmente, verificaremos se sistema proposto é totalmente adequado ao SAID. É também nosso objetivo, desenvolver uma função de avaliação da aprendizagem para substituir, mesmo que opcionalmente, a avaliação humana, de maneira que possamos implementar um conjunto de parâmetros de restrições auto-ajustáveis, utilizando diferentes abordagens (estatística ou mesmo redes neurais). A função de avaliação e os parâmetros ajustáveis permitirão, certamente, o refinamento contínuo da aprendizagem do sistema.

Uma outra possibilidade, é implementar uma opção para que o sistema possa aprender também utilizando um algoritmo incremental. Esta opção é certamente interessante e pode ser facilmente implementada, já que a base de conhecimento utilizada pelo S3O para gerar os fatos majoritariamente válidos não é descartada após a aprendizagem, mas só após a apresentação de uma nova amostra.

Tão logo as avaliações do S3O sejam concluídas, intensificaremos a nossa contribuição para a especificação do SAID junto aos trabalhos de Nóbrega [NÓBREGA 1998] e Ferneda [NÓBREGA 1998]. Essa contribuição deverá incluir a parte de geração de hipóteses para o conceito, que envolve a parte indutiva desse tipo de aprendizagem.

Finalmente, estaremos construindo uma especificação do nosso sistema em uma Rede de Petri [MARSAN 1993], utilizando a ferramenta G-CPN [FIGUEIREDO 1997]. Logo em seguida, o S3O deverá ser implementado em uma linguagem ainda não definida (estamos estudando a possibilidade de utilizar Java ou Jade) e então deverá ser disponibilizada para uso nos projetos em que se aplique.

Bibliografia

[AIM 1997] A.I.M. Artificial Intelligence Magazine - Fall issue 1997.

[ASIMOV 1994] Asimov, Isaac. **"I, Robot"** Mass Market Reprint edition (July 1994) Bantam Books; ISBN: 0553294385.

[BARBOUX 1990] Barboux, C., Sallantin, J. **"Contrôle par Objection dúne Théorie Semi-Empirique"** RIA Vol. 4-2, páginas 29-41.

[BELLMAN 1978] Bellman, R. **"An Introduction to Artficial Intelligence - Can Computers Think?"** San Francisco - CA. Boyd & Fraser - 1978.

[BEZERRA 1995] Bezerra, Haroldo C. F., **"Um Estudo Comparativo entre Algoritmos Indutivos Incrementais e Não-Incrementais Utilizados na Geração de Bases de Conhecimento a Partir de Exemplos"**, Tese de Mestrado em Inteligência Artificial, DSC/UFPB. 07.12.95 - Campina Grande, PB.

[BONE 1995] Bone, S. and Castro, M. **"A Brief History of Quantum Computation"** , Surprise'97, vol. 4. Inc. New York, 1995.

[BRACKER 1997] Backer, H., **"Wake up to Quantum Coffee"**. New Scientist Magazine, 15/03/1997.

[BRASSARD 1997] Brassard, G. **"Searching a Quantum Phone Book"**. Science Magazine, Vol 275, 31/01/ 1997.

[BUCHANAN 1978] Buchanan, B. G. 6 Feigenbaum, E. A. **"DENDRAL e Meta-DENDRAL", Their Applications Dimention"**. AIM N° 11, ano 1978, páginas 5-24.

[BUCHANAN 1982] Buchanan, B. G. **"Mechanizing the Search for Explanatory Hipotheses"** Philosophy of Science Association PSA Magazine 1982 V, páginas 129-146. Asquith & Nickels Editors, East lansing - Michigan.

[CARBONELL 1986] Carbonell, J. e Hood, G. **"The World Modelers Project: Objectives and Simulating Architecture"**. Em [Michalski 1986b].

[CARBONELL 1989] Carbonell, J. G. **"Introduction: Paradigms for Machine Learning"**. Artificial Intelligence. Special Volume on Machine Learning. Ed. Carbonell, J.G. Vol 40 N° 1-3, 09/1989 páginas 1-10, North-Holland.

[CARVALHO 1998] André Ponce de Leon F. de Carvalho em http://www.icmsc.sc.usp.br/~andre/gene1.html#intro.

[CHARNIAK 1985] Charniak, E., McDermott, D. **"Introduction to Artificial Intelligence"**, Addison-Wesley Publishers, Reading, Massachussets, 1985.

[CHONG 1994] Yu, Chong Ho. **"Abduction? Deduction? Induction? Is there a Logic of Exploratory Data Analysis?"** Anais do "Annual Meeting of American Educational Research Association", New Orleans, LA, 1994.

[CLARK 1994] Clark, Arthur C., **"2001 : A Space Odyssey"** - G K Hall Perennial Large Print Book - reprint edition 12/1994 - G K Hall & Co; ISBN: 0816174865 - 1994.

[COEN 1994] Coen, M. SodaBot: **"A Software Agent Environment and Construction System"**.06/1994 em: **http://www.cl.cam.ac.uk/users/rwab1/doc/**sodabot94.ps. gz.- 1994.

[COHEN 1982] Cohen, P. R e Feigenbaum, E.A. **"The Handbook of Artificial Intelligence"**. Vol 3, Kaufmann Editors, Los Altos, Ca. 1982.

[COHEN 1994] Cohen, P. R. **"An Open Agent Architecture"**. Anais do AAAI Spring Symp.: Software Agents. AAAI Press. Cambridge, Mass., 1994, páginas 1-8; 1994.

[CONVENEY 1995] Conveney, P. and Highfield, R **"Frontiers of Complexity"**. Ballantine Books, Random House, 1995.

[CURTIS 1975] Curtis, Edward S. **" The Sioux and the Apsaroke"**. Ed. Stuart Zoll. nos volumes III e IV de "North American Indians". New York: Harper and Row. 1975.

[DARDEN 1997] Lindley Darden (darden@umiacs.umd.edu) Paper **"Comitê História e Filosofia de Ciência"**, Departamento de Filosofia - U of Maryland, MD 20742 E.U.A.; 1997.

[DAVIDSON 1981] D. Davidson **"The Material Mind"**, em **[HAUGELAND 1981]** Páginas 339-354; 1981.

[DAVIS 1983] Martin D. Davis e Elaine J. Weyuker **"Computability, Complexity and Languages"**, Orlando, FL: Academic Press; 1983.

[DEJONG 1986] Dejong, J. F. e Mooney, R. **"Explanation-based Learning: An Alternative View"**. Em **[LI 1994]** página. 11.

[DELGRANDE 1987] Delgrande J. P. **"Formal Bounds on the Automatic Generation and Maintenance of Integrity Constraints"**. Proc. 6th ACM Symposium on Principle of Database Systems; 1987.

[DEUTSCH 1992] Deutsch, D. **"Quantum Computation"**. Physics World Magazine, June, 1, 1992.

[DIETTERICH 1986] Dietterich, T. G., and Michalski, R. S., (1986). **"Learning to Predict Sequences"** em **[MICHALSKI 1986A]**.

[DREYFUS 1981] H. L. Dreyfus **"From Micro-Worlds to Knowledge Representation: AI at an impasse"** em **[HAUGELAND 1981]**. Páginas 161-204.

[EKERT 1993] Ekert, A, **"Quantum Keys for Keeping Secrets"**. New Scientst Vol. 137 Jan. 16, 1993.

[ETZIONI 1994] Etzioni, O.; Weld, D. S. **"A Softbot-Based Interface to the Internet"**. Comm. ACM. 37(7). pp. 72-76. 06/1994.

[ETZIONI 1995] Etzioni, O.; Weld, D. S. **"Intelligent Agents on the Internet: Fact, Fiction, and Forecast"**. IEEE Expert. 06/1995.

[FEIGENBAUM 1963] Feigenbaum E. A. e Feldman, J. **"Computers and Thought"**. New York, NY: McGraw Hill; 1963.

[FERNEDA 1992] Ferneda, E.; Py, M.; Reitz, Ph.; Sallantin, J **"L'agent rationnel SAID: une application en géométrie"**, Proceedings of the First European Colloquium on Cognitive Science, pp. 175-192, Orsay (França), 1992.

[FERNEDA 1994] Ferneda, E.; Silva, C.A.P.; Teixeira, L.M.; Silva, H.M., **"A System for Aiding Discovery in Musical Analysis"**, Anais do I Simpósio Brasileiro de Computação e Música, Caxambu, 1994.

[FERNEDA 1995A] Ferneda, E.; Souza e Silva, M.E.; Silva, H.M., **"A system for aiding discovery: mechanisms for knowledge generation"**. Advances in Artificial Intelligence. 12th Brazilian Symposium on Artificial Intelligence. Lecture Notes in Artificial Intelligence, Vol. 991, Springer, Berlin, 1995.

[FERNEDA 1995B] Ferneda, E.; Souza e Silva, M.E.; Silva, H.M., Gonçalves, C.A.V., **"Elements for designing a system for aiding discovery"**. Proceedings of the XV International Conference of the Chilean Computer Science Society, pp. 210-221, Arica (Chile), 1995.

[FIGUEIREDO 1997] Figueiredo, J.A., D.D.S. Guerreiro, Perkusich, A., **"Modeling a Cooperative Environment Based on an Object-Modular Petri-net"** Proceedings of the IX International Conference on Software and Knowledge Engineering, Madrid, 1997.

[FINDLER 1979] Findler, N. V. **"Associative Networks: Representation and Use of Knowledge by Computer"**. New York, Academic Press - 1979.

[FULTON 1998] Roger Fulton, John Gregory Betancourt **"The Sci-Fi Channel Encyclopedia of TV Science Fiction"** December 1998; Batan Books, ISBN: 0446674788.

[GARDNER 1983] Gardner, Howard. **" Frames of Mind: The Theory of Multiple Intelligences"**. NY: Basic Books, 1983.

[GARDNER 1993] Gardner, H. **"Creating Minds: An Anatomy of Creativity as Seen Through the Lives of Freud, Einstein, Picasso, Stravinsky, Eliot, Graham, and Gandhi"**. NY: Basic Books, 1993.

[GASCUEL 1998] Gascuel, A. **"A Conceptual Regression Method"** Proceedings of the EWSL-'89 (European Working Section on Learning), december, 1989.

[GENESERETH 1994] Genesereth, M., and Ketchpel, S. **"Software Agents. In Communications of the ACM"**. 37,7 (July 1994). ACM, New York, New York: 48-53.

[GOULD 1983] Gould, Stephen jay. **"The Mismeasure of Men"** - Penguin Books, 1983. ISBN 0-393-03972-2.

[GREIF 1994] Greif, I. Desktop **"Agents in Group-Enabled Products"**. Em Communications of the ACM. 37,7 (July 1994). ACM, New York, New York: 100-105.

[HAUGELAND 1981] J. Huageland **"Mind Design: Philosophy, Psychology, Artificial Intelligence"** Mit Press, Cambridge, Massachussets.

[HEDBERG 1995] Hedberg, S. R. **"Intelligent Agents: The First Harvest of Softbots Looks Promising"**. IEEE Expert. August 1995.

[HOFFMAN 1997] Hoffman **"Is There a Logic of Abduction?"** Paper apresentado no 6° congresso da IASS-AIS, International Association for Semiotic Studies - Guardalajara, México - 06 /1997.

[HOROWITZ 1987] Horowitz, E., Sahni, S., **"Fundamentos de Estruturas de Dados"** Editora Campus, Rio de Janeiro, 1987.

[KAUTZ 1994] Kautz, H., Selman, B., Coen, M. **"Bottom-Up Design of Software Agents"**. In Communications of the ACM. 37, 7 (July 1994). ACM, New York, New York: 143-145.

[KNOBLOCK 1994A] Knoblock, C. A.; Arens, Y. **"An Architecture for Information-Retrieval Agents"**. Working Notes of the AAAI Spring Symp.: Software Agents. AAAI Press. Cambridge, Mass., 1994, pp. 49-56.

[KNOBLOCK 1994B] Knoblock, C. A.; Levy, A. **"Efficient Query Processing for Information Gathering Agents"**. CIKM Workshop on Intelligent Information Agents. 3 International Conference on Information and Knowledge Management. National Institute of Standards and Technology. Gaithersburg. Maryland. 1994.

[KUHN 1962] Kuhn, T. **"The Structure of Scientific Revolutions"**. Chicago, IL: University of Chicago Press.

[KULKARI 1988] Kulkari, D. e Simon, H.A. **"The Process of Scientific Discovery: The Strategy of Experimentation"**, Cognitive Science Magazine,N° 12: páginas 139-175, 1988.

[KURZWEIL 1990] Kurtzweil, R. **"The age of intelligent Machines"** The MIT Press - Cambridge, Massachussets, 1990.

[LAKATOS 1976] Lakatos, I. **"A lógica do descobrimento matemático: Provas e Refutações"**, Zahar Editores - Rio de Janeiro, RJ. 1976.

[LANGLEY 1987] Langley, P. , Simon, H. A., Bradshaw, G.L. & Zytkow, J.M. **"Scientific Discovery: Computational Explorations of the Creative Process"**. Cambridge Mass, MIT Press. 1987.

[LEBOWITZ 1980] Lebowitz M. "Generalization and Memory in an Integrated Understanding System". New York, NY: Yale University PhD Dissertation.

[LEBOWITZ 1983] Lebowitz, M. **"RESEACHER: An overview"** American Association for Artificial Intelligence, Proccedings of the National Conference in Artificial Intelligence - Los Altos - CA. William Kaufman Inc. Páginas 232-235.

[LEEUWEN 1990]. J. Van Leeuwen **"Algorithms And Complexity - Handbook of Theoretical Computer Science"**. The MIT Press, Cambridge - Mass. 1990.

[LEVY 1994] Levy, A. Y.; Yehoshua, S.; Srivastava, D. **"Towards efficient information gathering agents"**. Working Notes of the AAAI Spring Symp.: Software Agents. AAAI Press. Cambridge, Mass., 1994, pp. 64-70.

[LI 1994] Heng Li **"Machine Learning of Design Concepts"**, Southhampton: Computational Mechanics Publications, Hobbs the Printers, 1994.

[LINDSAY 1980] Lindsay, R. K., Buchanan, B.G.,Ffeigenbaum, E.A. e Lederbergt, J. **"Aplications of Artificial intelligence for Organic Chemistry - The DENDRAL project"** -Mc. Graw-Hill, New York, NY 1980.

[LINDSAY 1993] Lindsay, R. K., Buchanan, B.G., feigenbaum, E.ª e Lederbergt, J. **"DENDRAL: A Case study of the first Expert system for Scientific Hipothesis Exploration"**. AIM N° 61, páginas 209-261, 1993.

[LIQUIÈRE 1990] Liquière, M.; Sallantin, J., **"INNE (Induction in Networks): A structural learning algorithm for noisy exemples"**, Proceedings of the Fourth European Working Session on Learning (Montpellier - 1989), Pitman • Morgan Kaufmann Publishers Inc., Londres (Inglaterra), 1990.

[LIQUIÈRE 1992] Liquière, M., **"Recherche de structures dans des données bruitées"**, 1ères Journées Francophones d'Apprentissage et d'Explications des Connaissances, Dourdan (França), 1992.

[LUCAS 1997] Lucas. George, Glut, Donald F., Kahn, James **"The Star Wars Trilogy : Star Wars, the Empire Strikes Back, Return of the Jedi"** - Mass Market Paperback (February 1997) Del Rey ISBN: 0345384385.

[LUGER 1993] Luger, G. F. Stubblefield, W.A. "**Artificial Intelligence: Structures and Strategies for Complex Problem Solving**" Benjamin Cummings Publishers, Redwood City, CA, 1993.

[MAES 1994] Maes, P. "**Agents that Reduce Work and Information Overload**" Em Communications of the ACM. **37**,7 (July 1994). ACM, New York, New York: 31-40.

[MCCARTHY 1968] McCarthy, J. "**The Advice Taker**", em [MINSKY 1968].

[MCDERMOTT 1983] John P. McDermott: "**The Knowledge Engineering Process**". Database Engineering Bulletin 6 (4): 30-37 (1983).

[MICHALSKI 1983] Michalsky R.S., Carbonell J.G. e Mitchell T. "**Machine Learning Vol I**" - Los Altos, CA: M. Kaufmann Publishers.

[MICHALSKI 1983] Michalski, R. & Stepp, R.. "**Learning From Observation: Conceptual clustering**". Em [MICHALSKI 1986A].

[MICHALSKI 1986A] Michalsky R.S., Carbonell J.G. e Mitchell T. "**Machine Learning Vol II**" - Los Altos, CA: M. Kaufmann Publishers.

[MICHALSKI 1986B] Michalsky R.S., Carbonell J.G. e Mitchell T. "**Machine Learning, A Guide to Current Research**" - Los Altos, CA: M. Kaufmann Publishers.

[MINSKY 1968] Minsky, M. "**Semantic Information Processing**" - Cambridge, MS: MIT Press.

[MINSKY 1985] Minsky, M. "**Society of Mind**" - Technical report, Dept. of Computer Science, MIT, MIT Press.

[MITCHEL 1977] Mitchel T. M. "**A Candidate Elimination Approach to Rule Learning**" - Cambridge, Mass: Procs. Fifth International Joint Conference on Artificial Intelligence.

[MONGIOVI 1995] Mongiovi, G. "**Uso de Relevância Semântica na melhoria da qualidade dos resultados gerados pelos métodos indutivos de aquisição de conhecimento a partir de exemplos**"; Tese de Doutorado em Engenharia Elétrica, COPELE - DEE/UFPB, Campina Grande, PB, em 1995.

[MONTEIRO 1978] Monteiro, L. H. Jacy. "**Elementos de Álgebra**" - RJ, Livros Técnicos e Científicos Editora S/A.- Rio de Janeiro, 1978.

[NEWELL 1963] Newell, A. e Simon, H. A. "**GPS, a program that simulates human thought**" - Em [Feigenbaum e Feldman 1963], pgs 279-293.

[NGUIFU 1990] Liquière M., Mephu Nguifu E., "**LEGAL - Learning With Galois Lattice**" - Proceedings of the 5th JFA - páginas 93-113.

[NGUIFU 1994] E. Mephu-Nguifu "**Galois Lattice: A framework for Concept Learning - Design, Evaluation and Refinement**", Páginas 461-467, Proceedings do 6° TAI (Tools in Artificial Intelligence", New orleans, 1994.

[NÓBREGA 1997] Nóbrega, G.M.; Ferneda, E.; Figueiredo, G.C.A., "**A System for Aiding Discovery Based in a Multi-Agent Architecture: The communication modeling**", World Multiconference on Systemics, Cybernetics and Informatics - SCI'97, Special Session on Learning Control, Caracas, 1997.

[PASCOE 1998] Pascoe, Elaine., Kurtis, Bill. "**Animal Intelligence : Why Is This Dolphin Smiling**" - The New Explorers 1st edition - 06/1998- Blackbirch Marketing; ISBN: 1567112269.

[PERROSE 1991] Penrose, Roger. "**The Emperor's New Mind: Concerning Computers, Minds And Yhe laws of Physics**" - Penguin Eds - USA. ISBN 01-4014-534-6, 1991.

[QUILLIAN 1966] M.R. Quillian "**Semantic Memory**" - AFCRL-66-189 Ph.D. thesis. Pittsburgh: Carnegie-Mellon Univ. USA - 1966.

[QUINLAN 1986] Quinlan, J. R. "**Induction of Decision Trees**". Em "Readings in Machine Learning", Morgan Kaufmann -1990, Jude W. Shavlik and Thomas G. Dietterich editors.

[RICH 1991] Elaine Rich e Kevin Knight "**Artificial Intelligence**" - New York, NY: McGraw Hill 1991.

[RICH 1997] Elaine Rich, Kevin Knight e Janet Finlay "**Machine Learning**" Review by Dewan Shaju Khan made by reading through various AI books /other journals & computer magazines.

[RIECKEN 1994] Doug Riecken "**An Architecture of Integrated Agents**". ACM Computer Surveys (CACM) Vol. 37 :páginas 106-116, 146 (1994).

[RODDENBERRY 1992] Roddenberry. Gene, Meyer. Nocholas, Barr. Mike, "**Best of Star Trek**" - Michael Hill (Editor) August 1992, Diamond Comic Dist. Star Sys.; ISBN: 1563890097.

[RUSSEL 1995] Russel, S. Norvig, P. "**Artificial Intelligence: A Modern Approach**" Prentice Hall Inc. 1995. ISBN 0-013-103805-2

[SALLANTIN 1991A] Sallantin, J.; Quinqueton, J.; Barboux, C.; Aubert, J.-P., "**Tréories semi-empiriques: éléments de formalisation**", Revue d'Intelligence Artificielle, Vol.5, França, 1991.

[SALLANTIN 1991B] Sallantin, J.; Szczeciniarz, J.-J.; Barboux, C.; Lagrange, M.-S.; Renaud, M., "**Tréories semi-empiriques: conceptualization et ilustrations**", Revue d'Intelligence Artificielle, Vol.5, 1991.

[SCHALKOFF 1990] Schalkoff, R. J. "**Artificial Intelligence: An Engineering Approach**" McGraw-Hill editors, New York, NY 1990.

[SEARLE 1981] J.R.Searle "**Mind, Brains and Programs**". M.D. páginas 282-306.

[SHAFTO 1997] Shafto, M., Langley,P. "**Proceedings of the Nineteenth Annual Conference of the Cognitive Science Society**". Shafto & Langley (Eds.). Mahwah, New Jersey: Lawrence Erlbaum, 1997, pp. 161-166.

[SIMON 1977] Simon, H. A. "**Models of Discovery**" Models of Discovery - Paperback Issues Series - 1977 Kluwer Academic Publishers; ISBN: 902770970X.

[SIMON 1983] Simon, H. A. "**Machine Learning**" - Em [MICHALSKY 1983].

[SOWA 1984] Sowa, J. F. "**Conceptual Structures**". Addison Wesley Publishers - 1984

[TARPY 1978] Tarpy, R. M. "**Foundations os Learning and Memory**" - Glenview, IL: , Scott, Foresman & Company Publishers.

[TURING 1950] Turing, A. "**Computing Machinery and Intelligence**" Em [**FEIGENBAUM 1963**], páginas 11-35.

[VAUCLAIR 1987] Jacques Vauclair "**Animal Cognition : An Introduction to Modern Comparative Psychology**" 06/1996 - Harvard University Press; ISBN: 0674037030.

[VERGARA 1990] Vergara, W. H. "**Resolução de Problemas Baseados no Conhecimento Humano: Contribuições da Psicologia e de Inteligência Artificial à Ergonomia Cognitiva**" - Dissertação UFSC,Florianópolis 1990.

[WIELINGA 1990] Wielinga B., Boose J., Gaines, B. Screiber G. Someren M. Van (eds). "**Current trends on knowledge acquisition**" - Proceedings of tthe European Knowledge Acquisition Workshop, 1990.

[WILLE 1982] Wille, R. "**Restructuring Lattice Theory**" Symp. Ordered Sets Proceedings, Boston, Mass. páginas 445-470, 1982.

[WILLE 1991] Wille, R. "**Local Completeness of Conceptual Knowledges systems**", Symbolic-Numeric Data Analisys and Learning Proceedings, Versailles, 09/1991. Páginas 347-356. Diday & Lechevallier Eds.

[WILLE 1992] Wille, R. "**Concept Lattices and Canceptual Knowledges Systems**", Computer Math. Applications Vol. 23, Nº 6-9, páginas 493-515, 1992.

[WINSTON 1975] Winston,P. H. "**Learning Structural Discriptions from Examples, The Psychology of Computer Vision**", New York, NY: McGraw Hill.

[WINSTON 1992] Winston, P.H. "**Artificial Intelligence**" Addison-Wesley Publishers, Reading, Massachussets, 1992.

[WOODS 1983] **Woods, W.A. "Under what conditions can machine use symbols with meaning?"**, Proceedings of the IJCA (Iternational Conference on Artificial Intelligence - 1983), Kaufmann editors - Los Altos, CA, 1983.

[WOOLDRIDGE 1995] Wooldridge, M., Jennings, N. "**Intelligent Agents: Theory and Practice**" - At URL http://www.doc.mmu.ac.uk/STAFF/mike/ker95.ps.

[WOOLDRIDGE 1995] Wooldridge, M.; Jennings, N. R. "**Intelligent Agents: Theory and Practice**" - Knowledge Engineering Review. October 1994. Revised January 1995.

[ZYTKOW 1987] Zytkow, J. M., Langley, P., & Simon, H. A. "**Computer system of discovery STAHL**". Studia Filozoficzne or Zagadnienia Naukoznawstwa, 23, 518-536.