Agenor de Sousa Martins

Computação Baseada em Casos: Contribuições Metodológicas aos Modelos de Indexação, Avaliação, *Ranking*, e Similaridade de Casos

Tese submetida à Coordenação de Pós-Graduação em Engenharia Elétrica do Centro de Ciências e Tecnologia da Universidade Federal da Paraíba – Campus II, como parte dos requisitos básicos para a obtenção do grau de Doutor em Ciências no domínio da Engenharia Elétrica.

Área de Concentração: Processamento da Informação

Edilson Ferneda
(Orientador)

Campina Grande, Paraíba
2000

M386c Martins, Agenor de Sousa
Computacao baseada em casos : contribuicoes metodologicas aos modelos de indexacao, avaliacao, ranking, e similaridade de casos / Agenor de Sousa Martins. - Campina Grande, 2000.
185 f.

Tese (Doutorado em Engenharia Eletrica) - Universidade Federal da Paraiba, Centro de Ciencias e Tecnologia.

1. Raciocinio Baseado em Casos 2. Similaridade de Casos 3. Computacao Inteligente 4. Tese - Engenharia Eletrica I. Ferneda, Edilson II. Universidade Federal da Paraiba - Campina Grande (PB) III. Título

CDU 681.3(043)

COMPUTAÇÃO BASEADA EM CASOS: CONTRIBUIÇÕES METODOLÓGICAS AOS MODELOS DE INDEXAÇÃO, DE AVALIAÇÃO, DE RANKING, E DE SIMILARIDADE DE CASOS

AGENOR DE SOUSA MARTINS

Tese Aprovada em 03.07.2000

EDILSON FERNEDA, **Dr., UFPB**
Orientador

MARIA DE FÁTIMA QUEIROZ VIEIRA TURNELL, Ph.D., UFPB
Componente da Banca

ULRICH SCHIEL, Dr.rer.nat., UFPB
Componente da Banca

MARIA CAROLINA MONARD, Dr., USP-São Carlos
Componente da Comissão

GEBER LISBOA RAMALHO, Dr., UFPE
Componente da Comissão

NIZAM OMAR, Dr., ITA-SP
Componente da Comissão

CAMPINA GRANDE - PB
Julho - 2000

Agradecimentos

Agradeço a todos os colegas, funcionários e pós-graduandos com os quais conviví nesta ilha de conhecimentos (a Campina Grande do Campus 2) e dos quais, de algum modo, recebí encorajamentos. Devo mencionar, entre outros, Ulrich Schiel, Peter Nicolletti, Giuseppe Mongiovi, Hélio Menezes, Haroldo Catunda, Mário Ernesto, Joelson Carvalho, Jeorge Gomes, Josenilda Travassos (Secretária de Sistemas e Computação), Aninha Guimarães (Secretária do Mestrado de Informática), Ângela de Lourdes (Secretária do Programa de Doutorado - COPELE) e Rudra Dixit/Rohit Gheyi a quem agradeço pelo apoio em assuntos de programação Java.

Agradeço, em especial, a Edilson Ferneda (meu orientador de Tese) e a Bernardo Lula por terem oportunizado o pequeno sanduíche no DIAM – *Departement de Recherche en Informatique Automatique Mecatronique*, sob a liderança científica de Dr. Eugène Chouraqui, na Université Aix-Marseille 3, Marseille, France. Também a Fábio Paraguaçú, meu cicerone durante a temporada no DIAM.

Finalmente, agradeço a Elcí, Andréa e Rafael que, pacientemente, ficaram (quase) sem marido e sem pai por quase um longo tempo.

Este trabalho teve o suporte financeiro do PICDT – o Programa Federal de Capacitação de Docentes – gerenciado pela CAPES/MEC, o qual tornou possível a presente investigação sobre a computação baseada em casos, como paradigma da inteligência computacional.

Resumo

Percepção é a capacidade através da qual os agentes se apropriam de informações sobre o mundo no qual habitam esses agentes. Perceber através de similaridades e de analogias, em particular, constitui uma das formas mais avançadas de percepção, como demonstram as diversas escolas de abordagem da analagia; constitui um dos aspectos mais fundamentais da cognição humana. Computacionalmente, a similaridade é o fundamento sobre o qual repousa todo o formalismo do *Raciocínio Baseado em Casos* (RBC), como um dos ramo da Inteligência Artificial ou da computação inteligente. Em sua origem e em sua natureza, a tecnologia do raciocínio baseado em casos é afetada pela similaridade em todos os seus aspectos operacionais.

A tese aqui detalhada tem como objetivo fundamental explorar esse paradigma computacional do raciocínio baseado em casos, sob os seus aspectos da similaridade e de seus efeitos. O ponto de partida está no reconhecimento da *insuficiência* das abordagens de similaridade, em geral, e, em particular, da similaridade em RBC – dominada pelas métricas euclidianas e por enfoques estatísticos do tipo "vizinho mais próximo". Propomos e desenvolvemos nesta tese a aplicação da *teoria* de similaridade de Tversky-Gati ao RBC tendo como meta estender os tratamentos computacionais da indexação, da valiação, do *ranking*, da similaridade e de outros aspectos da computação baseada em casos. Nós experimentamos com esta similaridade baseada em teoria e, também, com as suas implicações, em um domínio do mundo real como a *análise empírica* do crédito financeiro.

Abstract

Perception provides agents with information about the word they inhabit. To perceive by using similarities and analogies is one of the most advanced forms of perceiving. It constitutes one of the most fundamental aspects of human cognition. Computationally, similarity is the corner-stone upon which relies the entire case-based reasoning technology (CBR), as an intelligent computing paradigm. In its origin and nature, the CBR technology is affected by similarity in all over its operational aspects.

The thesis here detailed has the general objective of exploring the CBR computational paradigm regarding to the similarity point of view and its implication for these paradigm processes. The start point, is the recognition of insufficiencies of current similarity approaches, in general, andparticularly, in the CBR domain where the CBR processes are dominated by euclidean metrics and statistical nearest-neighbours techniques. In this thesis, we propose to enhance the CBR methodologies by developing the application of Tversky-Gati *cognitive theory* to the CBR methodologies of case indexing, case evaluation, case similarity itself, case*ranking*, and case-based query-answering. We experiment with this computational extent of the cognitive theory in the empirical domain of*loan-underwriting* where a credit underwriter has to decide on recommending or rejecting credit applications based on their attribute similarities.

O teorema da incompletude de Gödel tem o sabor dos antigos contos de fadas que nos avisam que procurar o auto-conhecimento é embarcar numa viagem ... que ficará sempre incompleta

Hofstadter, D.R. *Gödel, Escher, Bach*: Basic Books, 1979, p. 697

Sumário

PARTE 2 – DESENVOLVIMENTO DAS METODOLOGIAS

Lista de figuras

Parte 1

Estado da arte da tecnologia de RBC

Qual o papel da Computação Baseada em Casos (Raciocínio Baseado em Casos) ao lado de outros paradigmas da Computação Inteligente? Os Capítulos 1 e 2 nesta Parte 1 procuram situar o RBC no campo da IA e também descrever os seus processos e metodologias respectivamente.

Introdução geral

Os achados mais importantes são os *métodos*.
Nietzsche, F. W., 1886 (Ao receitar o futuro da ciência)

0.1 Contexto do problema

Raciocínio Baseado em Casos (RBC) é um paradigma computacional de resolução de problemas que emprega um banco de dados ou *base de casos* de problemas anteriormente já resolvidos para solucionar um novo problema. O emprego desta base de casos implica, justamente, que os dados nela armazenados passem a representar episódios anteriores de resolução de problemas. Esta abordagem funciona então como um modelo a guiar o desenvolvimento de programas assistentes da resolução de problemas, e que acessam diretamente uma base de casos **[MAH 95, RIE 96]**. De uma maneira a mais compacta possível (e poucas tecnologias computacionais se prestam a uma tal compactação), o raciocínio baseado em casos pode ser expresso do seguinte modo:

> Para resolver um problema novo, relembre-se de um problema similar que você já resolveu no passado (um caso), e adapte a sua relembrada solução anterior, de tal modo a atender ao novo problema proposto.

Por conseguinte, a questão central inerente ao raciocínio baseado em casos vai consistir, pois, em levar o sistema a "relembrar" casos relevantes, reutilizar diretamente estes casos em uma nova solução ou adaptá-los se, para isto, se fizer necessário.

0.1.1 Estado da arte do RBC em síntese

Na condição de tecnologia computacional de solução de problemas, o RBC contrasta, por exemplo, com a abordagem dos sistemas expertos (*expert systems* – mas não – *specialist systems*). Ambas as abordagens repousam numa explícita *representação simbólica* de conhecimentos sobre como resolver um problema novo. Sistemas expertos usam a experiência passada ao armazenarem, numa base de conhecimentos, heurísticas generalizadas para assistirem na resolução de um problema. Essas heurísticas generalizadas podem ser armazenadas ou na forma de regras ou como inferências lógicas. O RBC, ao contrário, emprega a representação de episódios específicos de resolução de

problemas para aprender a resolver um novo problema. Ambas as abordagens fazem uso de experiências passadas para enfrentamento de novas situações. Porém, enquanto o RBC armazena essa experiência passada na forma de independentes episódios de resolução (uma espécie de *cache* semântico [**GOD 99**]), os sistemas expertos armazenam essa experiência na forma de regras heurísticas generalizadas.

Por exemplo, o desenvolvimento de um programa computacional para assistir analistas de crédito financeiro na concessão ou rejeição de uma solicitação de empréstimos (*loan-underwriting*) já foi recentemente tentado baseando-se em uma abordagem de sistemas expertos [**TAL 95**]. Nossos experimentos, por outro lado, desenvolvem uma abordagem de RBC para este mesmo problema. Sob o enfoque de sistemas expertos, procura-se codificar heurísticas sobre o caráter do proponente de crédito, sobre o histórico creditício do possível tomador e heurísticas capazes de identificar, financeira e patrimonialmente, um candidato a empréstimos. Já a abordagem de RBC se volta para a construção de uma base de casos de empréstimos previamente já decididos ou passados, de tal modo que, de entre aqueles casos passados um (ou mais) deles possa ser resgatado e selecionado para servir de ponto de partida para a análise de um novo empréstimo; um sistema computacional de casos de crédito nos evita de ter de "partir do nada" numa arriscada tarefa empírica tal como avaliar créditos. Ou seja: a abordagem de sistemas expertos, em geral, conduz à aplicação de *regras relevantes* em algum domínio de atuação [**MAH 95**, p. 3]. Do domínio do crédito, particularmente, estas regras relevantes objetivam definir os parâmetros de um novo empréstimos; enquanto que, na abordagem da tecnologia de RBC, um *caso relevante* de concessão de empréstimo passado vai ser resgatado e adaptado para aplicar-se a uma nova solicitação de crédito, por exemplo.

0.1.2 Problema em aberto: a similaridade de casos

Ora, reutilizar soluções passadas como ponto de partida para novas soluções só se torna possível em RBC em virtude de sua real natureza de *raciocíno por similaridade* e por analogia. O raciocínio analógico (ao contrário da *dedução*, *abdução* e *indução*) fundamenta-se, justamente, na idéia de que experiências e problemas que estejam sendo resolvidos possam se beneficiar de *insights* e da assistência de resoluções anteriores [**MAR 96a, MAR 96b**]. Através de uma analogia, podemos nos lembrar de um empréstimo/financiamanto já feito para um grande projeto de geração de energia elétrica cuja amortização dependia da taxa cambial – ITAIPÚ, por exemplo – ao termos de decidir, positiva ou negativamente, sobre um novo empréstimo para um outro projeto semelhante cuja amortização deva depender de juros de mercado. Usamos a analogia, freqüentemente, para explicar conceitos ou a razão de ser de nossas tomadas de decisão.

Ao enxergar a analogia da perspectiva da memória e da relembrança (*Teoria da Memória Huma-*

na), foi desenvolvido por Roger Schank o conceito de *organização de memória* ou de *pacotes de organização da Memória* (MOP), como um mecanismo para representações computacionais ou para a representação de experiências humanas em computadores (Schank, *Dynamic Memory*, 1982, In: **[KOL 93]**). É todo este trabalho nascido e expandido a partir dos estudos da analogia e da similaridade que desagua nesta inteira área de resolução de problemas baseada em Inteligência Artificial que é o RBC.

[A questão da similaridade] "afeta TODOS os aspectos do raciocínio baseado em casos" **[BAR 89]**. Mesmo assim, desenvolvimentos recentes, aplicações e as ferramentas de RBC que embutem métricas para a computação de similaridades – como facilidades para o usuário – parecem ignorar a natureza mesma da similaridade. Parecem ignorar a realidade de que a habilidade de perceber similaridades e analogias constitui um problema muito mais complexo e um dos aspectos mais fundamentais da *cognição* humana. E é este pressuposto cognitivo sobre a similaridade que deve guiar o seu processamento automático – para um confiável emprego desta similaridade em tarefas de: *(i)* reconhecimento; *(ii)* classificação; *(iii)* aprendizagem; e até mesmo em tarefas de *(iv)* descoberta científica e criatividade. Este constitui um dos problemas centrais ainda em aberto no RBC.

0.2 Formulação do problema da tese: similaridade baseada em teoria

O nosso problema a enfrentar consiste, portanto, na mensuração da *similaridade de casos* – não como um mero sub-produto menos nobre do RBC (encontrável em "prateleiras", em forma de métricas embutidas em ferramentas de RBC, por exemplo) – mas a mensuração com uma fundamentação. Nosso ponto de partida é o de que esta mensuração fundamentada fará aproximar, *qualitativamente*, a similaridade automática daquela similaridade conseguida pela cognição humana; uma tarefa que certamente vai requerer a adoção de uma *teoria* de percepção de similaridade ou a adoção de algum princípio de similaridade menos artificial (do que nas métricas puramente geométricas, por exemplo) e mais natural ou compatível com o modo humano de ver similaridades entre objetos e situações. Daí a demanda que vem sendo feita por métodos de similaridade mais fundamentados em "princípios" (*more principled methods*) desde o Workshop DARPA, ainda em 1989 **[BAR 89]**; uma demanda também reiterada por Janet Kolodner e outros cognitivistas (como Keith Holyoak e Paul R. Thagard, autores de *Mental Leaps*, 1995) ao defenderem que "*pesquisas sobre como as pessoas julgam a similaridade aqui são certamente relevantes*" **[KOL 91**, p. 16].

Investigar, em benefício da computação baseada em casos, um método de similaridade com suporte na experiência humana constitui, justamente, o problema central na presente investigação. Sobre a solução proposta para este problema particular vão repousar as demais metodologias aqui construídas e experimentadas.

0.3 Subproblemas investigados

Pesquisas em analogia e similaridade, iniciadas ainda 1995 **[MAR 96a, MAR 96b, MAR 97]**, nos conduziram de encontro ao "modelo de contraste" de Tversky-Gati (*contrast model*) **[TVE 78]** – de uma época em que o RBC simplesmente inexistia. O "modelo de contraste" constitui uma robusta formulação particular de como as pessoas percebem similaridade. A nossa investigação resgata este modelo teórico da área da ciência cognitiva (da *Categorização* e *Cognição* onde estava inserido) para operacionalizá-lo e estendê-lo – computacionalmente – de modo a nele introduzir aquela *computabilidade* necessária aos processos próprios do RBC.

A importação para o RBC e a introdução de computabilidade nesse modelo teórico de similaridade cognitiva nos permitiram a formulação e a resolução de problemas específicos de RBC, quais sejam:

(i) a indexação e representação de casos;

(ii) a similaridade de casos, propriamente;

(iii) o *ranking* flexível de casos decorrente da similaridade cognitiva;

(iv) a avaliação de casos do RBC;

(v) a modelagem de sistemas de indagação-resposta baseados em casos; e

(vi) a aplicação das soluções encontradas a um domínio do mundo real de interesse, computacionalmente.

Estes problemas investigados à luz da similaridade cognitiva, em síntese, são enunciados na seqüência.

0.3.1 Extensão computacional da similaridade tverskyana

Subproblema 1: *Como computar a similaridade entre casos de modo a incorporar significantes componentes qualitativos do modelo teórico de similaridade de Tversky-Gati?*

Aqui, o problema a resolver está em como impor requisitos cognitivos para se encontrar casos computacionais que mais realisticamente se emparelhem uns aos outros. Esta, como visto, é reconhecidamente uma tarefa para a qual se torna relevante examinar o modo como as pessoas fazem julgamentos de similaridade entre objetos, como asseverara J. Kolodner.

0.3.2 Casos como representação de objetos tverskyanos

Subproblema 2: *Como indexar/representar casos de modo apropriado à metodologia de computação de similaridades proposta acima?*

Este subproblema, portanto, é uma questão decorrente do subproblema anterior. Admitimos, por conseguinte, que a metodologia de computação da similaridade influencia o modo como os casos devam ser estruturados; influencia a metodologia de *indexação* e não vice-versa.

0.3.3 *Ranking* flexível de casos similares

Subproblema 3: *Como estender a metodologia de similaridade cognitiva para também obter-se o ordenamento ou o ranking flexível de casos?*

O modo convencional como o RBC realiza o *ranking* de casos padece daquilo que estamos a denominar de *tirania do ordenamento final*. Uma vez o algoritmo de *ranking* haver produzido o seu ordenamento dos casos resgatados, não haverá mais opção para o usuário dessa ordenação. O nosso problema aqui consiste em estender o mesmo enfoque da similaridade cognitiva para introduzir flexibilidade no *ranking* de casos, permitindo ao usuário ordenar casos conforme os "pontos de vista" desse usuário sobre as similaridades e seus parâmetros.

0.3.4 Avaliação de casos através de métricas

Subproblema 4: *Como avaliar uma base de casos, a partir da* QUALIDADE *dos resultados das consultas feitas por usuários?*

Antever-se que a participação do usuário de bases de casos seja vital neste processo de avaliação de sistemas baseados em casos – um problema que também afeta áreas da computação como Banco de Dados e *Information Retrieval*, por exemplo.

0.3.5 Indagação-Resposta baseada em casos

Subproblema 5: *Como integrar as soluções metodológicas propostas para os problemas 1, 2, 3, e 4 incorporando-as a um contexto mais amplo de preocupações do RBC?*

Em outros termos: como unificar as metodologias propostas orientando-as para um problema mais amplo do RBC? A meta em relação a esta indagação metodológica é identificar e formalizar uma aplicação que funcione como um "guarda-

chuva" para os modelos desenvolvidos.

0.3.6 Experimentação em domínio do mundo real

Subproblema 6: *Como experimentar, computacionalmente, com as soluções propostas para cada um dos sub-problemas identificados acima, não em um domínio "de brinquedo" mas em um domínio "de verdade"?*

Na extensão do modelo cognitivo tomado como base, a proposta consiste em concentrar os experimentos em um domínio do mundo real de claro interesse computacional, diferentemente do modelo de Tverski-Gati.

0.4 Contribuições principais da tese

As soluções encontradas e propostas para os problemas acima formam as principais contribuições das investigações realizadas. São contribuições que objetivam estender a área do RBC e se enquadram em duas categorias: Contribuição para a concepção da similaridade cognitiva de casos e Contribuições para as metodologias de processamento do RBC.

0.4.1 Contribuições para a concepção da similaridade cognitiva de casos

Estender o modelo cognitivo de similaridade de Tversky-Gati para a computação baseada em casos significa contribuir para o RBC em relação aos aspectos seguintes:

- *Similaridade fundamentada em teoria.* O RBC passa a empregar um mecanismo de similaridade não *ad hoc*, mas um mecanismo com uma reconhecida fundamentação teórica;
- *Plausibilidade cognitiva da similaridade.* A plausibilidade do modelo cognitivo está comprovada por experimentação específica da área cognitiva, por exemplo, através dos métodos de *estímulo perceptivo* e *estímulo semântico* [**TVE 78**, p. 85]. É de se esperar que esta plausibilidade também seja estendida, por herança, ao modelo computacional de similaridade dele resultante;
- *Similitude e dissimilaridade na mesma métrica.* Em oposição a outros mecanismos de comparação de casos que consideram ora a similitude de atributos, propriamente, ora as diferenças entre estes atributos (a exemplo do sistema CASEY – cf. Anexo A), a similaridade cognitiva pressupõe a estreita correlação entre similitude de atributos e também diferença de atributos. A métrica de similaridade desenvolvida acopla, em uma mesma formulação, tanto a similitude quanto a dissimilaridade de atributos;
- *Diagnosticidade de atributos.* Em oposição a outros mecanismos de comparação de casos

que trabalham apenas com a importância de valores de atributo (medida em termos de *pesos*), a similaridade cognitiva acopla ainda o conceito de *diagnosticidade* para expressar a essencialidade de atributos na especificação/definição ontológica de objetos/situações;

- *Integração de processos do RBC*. A importação do modelo de Tversky da área cognitiva para a área do RBC, finalmente, não constitui uma aquisição estanque ou restrita apenas à métrica de similaridade, em si. Ela permite a integração da metodologia da similaridade com os demais processos do RBC.

0.4.2 Contribuições para as metodologias de computação de casos

As contribuições metodológicas, propriamente, são seis, incluindo entre elas o próprio modelo de resolução do problema tomado para experimentações. Segue uma síntese destas soluções contributivas.

Solução 1: *SIM(m,p) – similaridade de casos baseada em cognição*

O ponto de partida das metodologias está na concepção e na explicitação para o RBC da métrica cognitiva de Tversky-Gati. Esta explicitação é providenciada de modo a dar conta da similaridade cognitiva tverskyana e precisa caracterizar um caso do RBC através da representação dos seguintes componentes: *(i)* atributos; *(ii)* diagnosticidade de atributo; *(iii)* valor de atributo; *(iv)* atributos compartilhados por casos; *(v)* atributos não compartilhados; *(vi)* importância de atributos compartilhados; *(vii)* importância de atributos não compartilhados.

Solução 2: *IBT – indexação de casos baseada em tabela*

Para acomodar todos estes componentes no interior dos casos, propõe-se um modelo de organização de índices denominado *Indexação Baseada em Tabela* (*IBT*), uma forma tabular para os casos sobre os quais se aplica a similaridade *SIM(m,p)*.

Solução 3: *ORDEN – ordenamento ou ranking flexível de casos*

A idéia básica nesta metodologia *ORDEN* é usar a própria métrica da similaridade (o seu *pré-processamento* na forma $aX - bY - cZ$) para gerar a discriminação dos casos a serem ordenados. Haverá tantos ordenamentos de um mesmo conjunto de casos quantos forem os pontos de vista do usuário sobre a similaridade buscada e sobre os seus parâmetros ($a, b, c \geq 0$).

Solução 4: *AVAL – avaliação qualitativa de casos baseada em métricas*

Sistemas baseados em casos acessam um espaço de casos e retornam um conjunto deles em resposta a uma indagação do usuário. Nem todas estas respostas, porém, costumam ser igualmente aceitas como úteis pelo usuário. A proposta então é recorrer ao mesmo método de avaliação através de métricas já em vigor na área de *Information Retrieval*.

Solução 5: *I-RBC – modelo de indagação-resposta baseado em casos*

Em relação à solução do subproblema 5, mostramos como o paradigma de *Query-Answering Systems* (representado sobretudo pela *Escola Escandinava*[1]) pode se constituir um paradigma aglutinador das soluções 1, 2, 3, e 4. Ou seja, mostramos como orientar para os modelos de indagação-resposta baseados em casos as metodologias de RBC aqui propostas .

Solução 6: *Experimentações em análise de crédito (loan-underwriting).*

Recorremos à área do crédito financeiro para efeito de exemplificações e experimentação. O interesse em tomar a análise empírica do crédito como área de experimentações decorre não apenas da importância das operações de crédito para quaisquer das atividades econômicas e produtivas mas decorre, sobretudo, das suas características próprias (a serem vistas) e do interesse que a computação inteligente vem demonstrando para com o domínio. Em nossas aplicações demonstramos, principalmente:

- Como casos de crédito passam a ser objetos manipuláveis que devem encerrar alguma resposta em matéria de concessão de um crédito pretendido e de sua qualidade. Ou seja: mostramos como encapsular no interior de casos aqueles componentes que venham a funcionar como *lições* (ou respostas) úteis sobre possíveis decisões de concessão ou não de créditos financeiros.
- Como casos de crédito passam a ser objetos cujas representações devem se enquadrar na forma geral do modelo *IBT* estabelecido para os objetos tverskyanos. Ou ainda: como, conjuntamente, levar em conta tanto aqueles atributos associados a casos de créditos a comporem bases de casos, quanto os votos ou pesos associados a esses atributos e seus respectivos valores.
- Como casos de crédito passam a ser objetos cujas similaridades também devem ser computadas segundo o modelo *SIM(m,p)* estabelecido para os objetos tverskyanos, em geral.

[1] Denominamos de Escola Escandinava a linha de pesquisa sobre Sistemas *Query-Answering* liderada por Troles Andersen, da Universidade de Roskilde, da Dinamarca.

0.5 Importância da tese

A importância dos trabalhos aqui introduzidos decorre, em parte, tanto da relevância das metodologias para os próprios processos fundamentais do RBC quanto do *nível de operacionalização* e integração das soluções apresentadas. Em parte também, estas contribuições são importantes em decorrência da própria importância do RBC como um todo. Sobre a importância do RBC, ver-se-á mais à frente a enorme aplicabilidade desta tecnologia (cf. Capítulo 1) o que leva Guilherme Bittencourt a concordar com Roger Schank ao comentar produções recentes sobre o RBC:

> "Esta tecnologia, e seu problema teórico fundamental, a INDEXAÇÃO de grandes bases de conhecimento, já haviam sido considerados por Schank como a mais complexa e mais promissora área da IA simbólica." [**BIT 98**, p. 326].

0.6 Estrutura da tese

A organização do presente documento tem uma estrutura linear de leitura – ou seja – a ordem dos capítulos oferece a melhor estratégia para o encadeamento de seus conteúdos. Seus conteúdos podem ser grupados em três partes distintas, como segue:

- Parte 1: *Estado da arte da tecnologia de raciocínio baseado em casos*. Compreende os capítulos 1, e 2, onde o primeiro situa o RBC no campo da IA e o segundo descreve as metodologias e os algoritmos atuais do RBC.
- Parte 2: *Desenvolvimento das metodologias*. Compreende os capítulos 3, 4, 5 e 6. O Capítulo 3 representa um esforço de integração de todas as propostas metodológicas desenvolvidas. O capítulo fornece uma visão geral (*overview*) nos termos da solução 5 já introduzida. Os capítulos 4, 5 e 6 detalham cada uma das metodologias integrantes desta visão geral.
- Parte 3: *Experimentos e comparações com os trabalhos relacionados*. Compreende o Capítulo 7, onde são discutidos os resultados computacionais dos modelos de similaridade, de indexação de casos e de *ranking* de casos.
- Finalmente, sintetiza-se os resultados alcançados, com ênfase em algumas questões ainda em aberto e a propositura de linhas de pesquisa delas decorrentes.

0.7 Conclusão

O presente capítulo constitui uma síntese, a mais fiel possível, de toda a investigação por nós realizada na área da computação baseada em casos. A estruturação deste capítulo introdutório foi organizada de modo a enfatizar os objetivos da investigação proposta (a similaridade baseada em teoria)

e todos os subproblemas decorrentes desse objetivo maior; como também de modo a ressaltar as soluções alcançadas para cada processo do raciocínio baseado em casos, objeto de investigação.

Parte 1

Estado da arte da tecnologia de RBC

Qual o papel da Computação Baseada em Casos (Raciocínio Baseado em Casos) ao lado de outros paradigmas da Computação Inteligente? Os Capítulos 1 e 2 nesta Parte 1 procuram situar o RBC no campo da IA e também descrever os seus processos e metodologias respectivamente.

Capítulo 1

RBC como paradigma da Inteligência Computacional

1.1 Introdução

Raciocínio Baseado em Casos (RBC) constitui uma tecnologia computacional – do campo da computação inteligente ou Inteligência Artificial (IA) – cujas aplicações estão cada vez mais a se diversificarem (o RBC, como *tecnologia*, está amplamente caracterizado, por exemplo, no texto de Mary Lou Maher, M. Bala Balachandran e Dong Mei Zhang, sobre o RBC aplicado ao *design* [MAH 95]). Trata-se de um recurso de modelagem computacional apropriado para soluções e respostas a problemas tais como:

- Problemas de um gerente de banco ao examinar o cadastro de um cliente: *Posso autorizar um empréstimo a este cliente? De quanto?*
- Problemas de um corretor de imóveis frente a um proprietário: *Qual o valor mais apropriado para este imóvel?*
- Problemas de um mecânico de automóveis ao diagnosticar falhas: *O que está causando este problema?*
- Problemas de um advogado ou jurista quando em disputas jurídicas: *Quem poderá ganhar esta causa?*
- Problemas de um médico frente a seu paciente: *Qual o diagnóstico para estes sintomas? Qual o tratamento mais indicado?*
- Problemas de um geólogo quando em prospecção petrolífera: *Com esta configuração de solo, existirá petróleo neste local?*
- Problemas de um operador de alto-forno siderúrgico em atividade de monitoração: *Considerando o tipo de liga dentro do alto-forno, devo aumentar a sua temperatura?*

Problemas semelhantes a estes costumam aparecer, basicamente, em tarefas de *classificação*, *planejamento* e *síntese*. Em tarefas de síntese, o interesse maior está na geração de artefatos (por

exemplo, a geração de circuitos através de descrições em lógica de primeira ordem, ou a geração, de programas computacionais especificando-os para um provador automático). Tarefas de planejamento, por outro lado, se caracterizam por requerem planos que, por sua vez, precisam ser apropriadamente postos em uma dada seqüência, de tal modo que as últimas etapas de um plano não venham a desfazer os resultados já obtidos em etapas anteriores. Por fim, as tarefas que envolvem classificação ou categorização se caracterizam pela necessidade de se enquadrar um certo objeto ou uma certa situação ou evento em uma dada categoria pré-determinada.

Mostramos, neste capítulo, tanto esse interesse industrial despertado quanto a relevância estatística que o RBC vem assumindo na resolução destas abrangentes classes de problemas (seções 1.2 e 1.3 seguintes). O capítulo, no todo, não somente revela este interesse industrial como explora ainda uma questão que nem sempre tem merecido a devida atenção qual seja: a questão sobre o enquadramento ou papel teórico do RBC dentro da IA. Posicionamos, portanto, o RBC no âmbito da inteligência computacional e de suas aplicações, o que se faz necessário antes mesmo que sejam analisados os seus processos internos e antes mesmo que sejam identificadas as principais dificuldades existentes nestes seus processos internos (objeto de estudo, particularmente, do Capítulo 2 seguinte).

1.2 Interesse industrial

Um dos aspectos mais significativos na rápida evolução da tecnologia de RBC é o crescente interesse industrial girando ao seu redor, o que já foi detectado por Janet Kolodner ainda em 1993. Ao preparar o seu livro texto – e até hoje a "biblia" do RBC – Kolodner havia levantado 75 importantes sistemas construídos com base nesta tecnologia **[KOL 93]**. Apenas, seis meses depois, a mesma Kolodner já se deparara com a descoberta de mais 30 novos sistemas desta categoria, levando a crer que estas aplicações continuam a crescer cada vez mais **[KOL 96]**. Dado o seu apelo intuitivo de tecnologia centrada na experiência (dos *casos* passados) e dada também a sua operacionalidade, o RBC vem se distingüindo como de interesse crescente para a indústria e para os grandes negócios ou corporações.

Os exemplos da NEC japonesa e da Lockheed americana são bem representativos na evolução da tecnologia de RBC. A NEC emprega o RBC em um sistema *help desk* para suporte a problemas de software. Uma biblioteca de 20.000 casos sobre *bugs* em software juntamente com um algoritmo de *matching* parcial formam o *kernel* do sistema SQUAD **[KOL 96]** e permitem aos seus depuradores de programas usar esta biblioteca como um valioso *banco de conhecimentos* sempre que se necessite de uma urgente solução para corrigir um novo *bug* no interior de seus inúmeros sistemas. A Lockheed, por outro lado, é detentora de importante aplicação industrial. Ela emprega o RBC na

modelagem de CLAVIER, um sistema que funciona como um assistente semi-automático de operações em autoclaves (gigantes alto-fornos industriais). O sistema ajuda na construção de aeronaves, em particular no carregamento ou *layout* em autoclave daquelas partes de aeronaves moldadas em materiais compostos a serem levados para depuração (como grafite e fibra de vidro).

Ao lado destas importantes aplicações, o interesse industrial pelo RBC também tem se refletido no aparecimento no mercado de inúmeras *shells* [**ALT 95**]. As *shells* em RBC são ferramentas específicas para o desenvolvimento de aplicações – análogas àquelas ferramentas já existentes para os sistemas baseados em regras.

1.3 Relevância das aplicações

As atividades industriais ou de negócios que vêm sendo beneficiadas com aplicações de RBC foram recentemente levantadas/estudadas pelo inglês Ian Watson (University of Salford, RU) [**WATSON 97**]. Teoricamente, as atividades identificadas como sendo mais apropriadas ao emprego do RBC têm sido as atividades constantes da Figura 1.1.

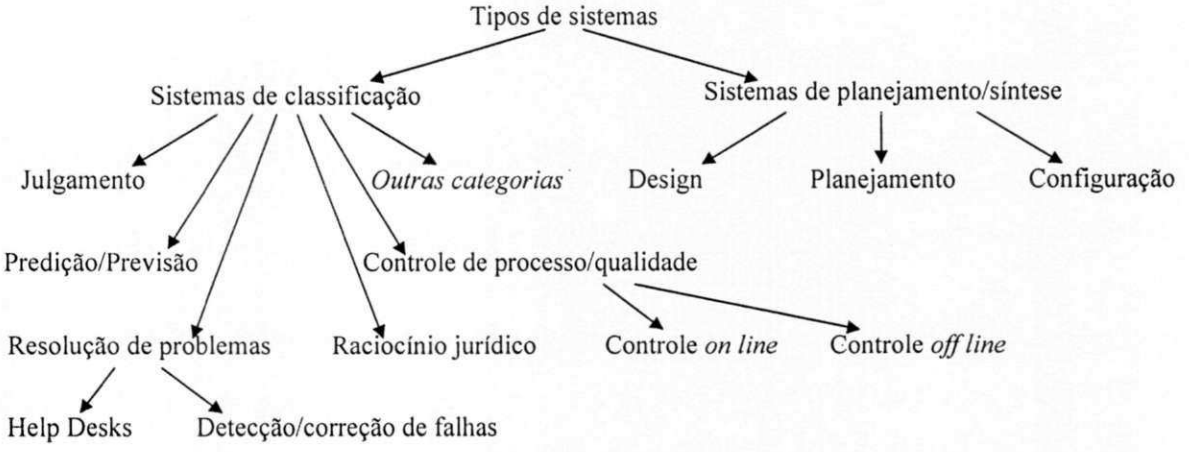

Figura 1.1: Aplicações do RBC em *classificação*, *planejamento* e *síntese*

Operacionalmente, o levantamento de Watson dá conta de que 58,5% de todas as aplicações da tecnologia de RBC estão concentradas no desenvolvimento de sistemas do tipo *help desk* que, em sentido amplo, designam os sistemas de apoio a clientes e usuários. Por outro lado, as aplicações em outras áreas de atividades tradicionalmente automatizáveis através de outros recursos da computação alcançaram 41,5%. Esta estatística, portanto, expressa a relevância da tecnologia de RBC na modelagem dos sistemas de apoio ao cliente ou ao usuário. Entre estes sistemas de apoio a cliente/usuário (ou seja, entre os 58,5% das aplicações, que corresponde a 79 aplicações levantadas) convém destacar o papel do RBC no desenvolvimento das seguintes classes de *help desk*:

(i) o apoio em matéria de *hardware* entra com 14 aplicações ou 17,7%;

(ii) o apoio em matéria de *software* entra com 10 aplicações ou 12,7%;

(iii) os sistemas de *apoio comercial* e de *consumo* registraram 10 aplicações ou 12,7%;

(iv) os sistemas *help desk* para *finanças* e *seguros* registraram 11 aplicações ou 17,7%;

(v) sistemas *help desk* construídos para *telecomunicações*, com 5 aplicações ou 6,3%;

(vi) *transporte* e *manufatura*, com 5 aplicações ou 6,3%;

(vii) *utilidades*, com 9 aplicações ou 11,4%;

(viii) *terceirização*, com 6 aplicações ou 7,6%; e

(ix) *outros sistemas*, com 9 aplicações ou 11,4%.

Mostrado, assim, o RBC como um *know-how* de ampla aplicabilidade industrial, tratamos, a seguir, de mostrar o seu enquadramento dentro da computação, particularmente dentro da computação inteligente.

1.4 Posicionamento do RBC na Inteligência Computacional

Qual o papel do RBC no contexto da IA? Qual a visão corrente de inteligência artificial que costuma vir associada ao emprego crescente desta tecnologia de casos? Do ponto de vista computacional, o RBC se enquadra teoricamente como um ramo importante da IA (Engenharia do Conhecimento, Teoria de Agentes), não obstante as suas fortes vinculações originais com a Lingüística e com as Ciências Cognitivas **[KOL 93]**.

Ora, observa-se na literatura que este relacionamento do RBC com a disciplina-mãe (a IA) nem sempre foi *clara e devidamente explicitado*, nem pela comunidade de RBC, particularmente, nem pela comunidade de IA, em geral. Daí a necessidade, neste ponto, de esclarecermos esta questão do posicionamento do RBC em relação à IA, uma vez que a meta geral em nossa pesquisa é a de poder contribuir para o desenvolvimento do RBC como paradigma da inteligência computacional. Três levantamentos a este respeito passam a ser analisados, na seqüência:

- Um levantamento baseado no texto básico de J. Kolodner que, tendo trabalhado na equipe de Roger Schank, foi também uma das criadoras do RBC;
- Um levantamento junto ao texto básico de Stuart Russell e Peter Norvig que difundiram a denominação de *IA moderna* e a concepção de IA como a ciência de agentes inteligentes;
- Um levantamento, por fim, baseado no artigo de Christopher Riesbeck – também um dos ex-

alunos de Schank criadores do RBC e que, em oposição a Russel e Norvig , enxerga um papel alternativo a ser exercido pelo RBC.

1.4.1 RBC e IA: Janet Kolodner (1993)

Tomemos, para exame desta importante vinculação, o texto – *Case-Based Reasoning*, de Janet Kolodner – uma das criadoras da tecnologia em apreço. Somente em três momentos a autora *explicitamente* relaciona o RBC com a IA neste longo texto básico sobre o estado da arte desta tecnologia. São, porém, relacionamentos estabelecidos ainda muito pontualmente. A autora ora apenas cita a IA, ora relaciona RBC e IA no que tange aos pontos de *aprendizagem*, *representação de conhecimento*, e *Engenharia de Conhecimento* como segue [**KOL 93**, pp. 7-8, p. 571]:

(i) *Aprendizagem*. Kolodner distingüe, muito claramente, a *aprendizagem* dentro do paradigma de RBC daquelas outras classes de aprendizagem viabilizadas em outros ramos da IA. A aprendizagem por RBC é obtida sobretudo através da "acumulação de novos casos e através da atribuição de novos *índices*" a estes casos. No resto da IA, no entanto, a aprendizagem usualmente significa *aprendizagem de generalizações*; generalizações que podem ser obtidas, quer através de processos indutivos (*aprendizagem indutiva*), quer através de processos baseados em explicação (*aprendizagem baseada em explicação*).

(ii) *Representação de Conhecimento*. Kolodner também estabelece a distinção entre a *representação de conhecimento* própria do RBC e aquelas outras classes de representação em outros ramos da IA (por exemplo, *modelos*, *árvores* e *regras*). Em RBC, os "casos representam conhecimentos específicos ligados a situações específicas e representam conhecimentos em um nível fundamentalmente operacional". No resto da IA, em geral, o modo como os desenvolvedores trabalham o conceito de conhecimento é diferente. Trabalha-se a representação de conhecimento buscando mecanismos que tornem esse conhecimento (a sua representação) o mais genérico possível e de tal modo a torná-lo aplicável o mais amplamente possível.

(iii) *Engenharia do Conhecimento*. Ao nomear claramente a *Engenharia do Conhecimento*, J. Kolodner refere-se à necessidade de desenvolvimento de ferramentas para esta engenharia e, por conseqüência, para o RBC. A autora, porém, não desenvolve qualquer esforço de enquadramento deste último dentro desta engenharia.

Portanto, não chega a ser analisado em J. Kolodner o papel amplo do RBC nas suas conexões com a IA; sequer o texto de Kolodner chega a fazer a opção por uma clara conceituação de IA (ver índice remissivo de [**KOL 93**]), como enfaticamente o fazem Russell e Norvig [**RUS 95**].

1.4.2 IA moderna: Stuart Russell & Peter Norvig (1995)

No trabalho de Russel e Norvig, claramente, foi feita uma opção pelo conceito de *agente inteligente* como temática unificadora do texto inteiro [**RUS 95**]. Aliás, a proposta destes pesquisadores é a de que os *agentes inteligentes* de todos as classes devam servir para unificar (não apenas o seu texto mencionado) mas todas as áreas científicas da IA. O conceito de agente inteligente constitui o elo unificador de toda a problemática explorada no texto e com base nesta visão eles definem a IA [**RUS 95**, p. vii]:

> [... O] problema da IA consiste em descrever e construir agentes que recebam PERCEPTS dos seus ambientes e realizem AÇÕES.

A conseqüência natural desta visão é que, em vez de continuar a conceber a IA como uma seqüência padronizada de temáticas e problemas (tais como busca, representação de conhecimento, inferência baseada em regras, etc), a moderna IA terá como objetivo amplo buscar um denominador comum para seu domínio científico. A proposição então é a de que sub-áreas como Robótica, Processamento de Linguagens Naturais, Visão por Computador, Processamento da Fala, Engenharia do Conhecimento, Programação Automática, Redes Neuronais, Algorítmos Genéticos e Computação Evolutiva, Resolução de Problemas e Aprendizagem de Máquina sejam todas elas áreas integráveis através do elo unificador dos agentes inteligentes.

O paradigma do raciocínio baseado em casos – sob esta visão de IA – tem pois o papel de servir como uma tecnologia de construção dos agentes inteligentes, muito embora este papel também não venha a ser explicitamente detalhado no citado trabalho de Russell e Norvig. Não pertencendo à comunidade de RBC, Stuart Russell e Peter Norvig apenas referenciam os trabalhos de Schank e Kolodner [**RUS 95**, pp. 23, 646], além de se referirem à relevância do RBC nos trabalhos de *planificação baseada em casos* desenvolvidos por Hammond [**HAM 89**].

1.4.2.1 RBC e ciência de agentes: questão do curto prazo × longo prazo

Adotar os agentes inteligentes autônomos como constituindo a meta fundamental da inteligência artificial e, *por conseqüência*, a meta do RBC, tem sido uma proposição a merecer críticas por parte de criadores do RBC tal como Riesbeck, um pesquisador da mesma estatura de J. Kolodner. Duas foram as críticas iniciais feitas por Riesbeck a essa concepção da IA como ciência dos agentes [**RIE 96**, p. 376]:

(i) Agentes inteligentes estão ainda tão distantes da realidade que não poderiam ainda ajudar em concreta tomada de decisão aqui e agora.

(ii) Agentes inteligentes nem mesmo seriam aquilo que haveria de se esperar dos computado-

res, em uma grande variedade de situações imediatas ou de curto prazo.

1.4.2.2 RBC e ciência de agentes: questão da autonomia de agentes

A crítica de Riesbeck se estende também à exigência de *autonomia* como propriedade para agentes. Em primeiro lugar, argumenta Riesbeck, a pressuposição de autonomia em agentes ignora uma das diferenças primordiais entre computadores e seres humanos. Programas computacionais costumam ser componentes tão dependentes e tão integrados a outros sistemas mais amplos, muito mais até do que costumam ser as próprias pessoas (exceto, metaforicamente falando). Rarissimamente, programas podem vir a se comportar como agentes independentes ou autônomos. Para exemplificar esta realidade, sejam considerados apenas dois exemplos: um computador de bordo no interior de um carro e um servidor qualquer em uma rede de computadores. Estes sistemas nunca serão autônomos por completo, segundo Riesbeck. Eles estarão a exigir *protocolos de comunicação* tão rígidos e serviços em tempo real tão exigentes, muito mais do que aquilo que hipotéticos agentes inteligentes autônomos poderiam vir a tolerar.

1.4.2.3 RBC e ciência de agentes: questão da indefinição evolutiva da IA

Agente inteligente como meta, além da questão relativa à autonomia destes agentes, acarreta também para a IA um compromisso de longo prazo, ainda muito distante para se concretizar. Tal compromisso de longo prazo, por sua vez, leva a uma clara indefinição sobre o que a IA deve fazer, no curto prazo; uma indefinição que também impossibilita orientar pesquisas e desenvolvimentos voltados para as necessidades do curto prazo. Nesta indefinição de longo prazo, todos os difíceis problemas de IA passam a ter a mesma importância. Como resultado disso, fica difícil saber, por exemplo, qual deverá ser o próximo melhor problema de IA que deve ser atacado, em razão de que a meta final de agentes fica a fugir de vistas. Daí a necessidade de uma nova compreensão e de um novo papel mais pragmático para a IA e para o RBC como preocupação dominante na pesquisa de Riesbeck.

1.4.3 IA pós-moderna: Christopher Riesbeck (1996)

Riesbeck atribui um papel para a IA que contrasta com os conceitos convencionais existentes tais como:

(i) IA como automação do comportamento inteligente **[LUG 93]**;

(ii) IA preocupada com máquinas fazedoras de coisas atribuídas a pessoas inteligentes **[MIN 89]**;

(iii) IA como estudo de faculdades mentais via modelos computacionais **[CHA 87]**; e

(iv) IA como ciência dos agentes inteligentes **[RUS 95]**.

A IA como sendo a ciência dos agentes inteligentes corresponde ao pensamento de Russell e Norvig (conhecido como a *moderna* IA). Christopher Riesbeck parte desta concepção de *moderna IA* proposta por Russell e que o leva inclusive a batizar a sua própria concepção de *IA pós-moderna.* Uma concepção que - assim como a da moderna IA - também está centrada na necessidade de uma idéia unificadora, qual seja a proposta da IA como *Engenharia de Componentes Inteligentes* - e onde o RBC possa ter o seu lugar de destaque. Com base nesta visão, Riesbeck estabelece que **[RIE 96**, p. 377]:

> [... O] problema fundamental (ou objetivo) da IA consiste em descrever e construir *componentes* de software que reduzam a ESTUPIDEZ daqueles sistemas nos quais esses componentes venham a funcionar.

Componentes, aqui, designam "pedaços isolados de software que podem se interconectar automaticamente via redes, aplicações, linguagens, ferramentas e sistemas operacionais" e cuja tecnologia promete alterar radicalmente a maneira como se desenvolve software **[CHE 96]**. Na origem desta conceituação de IA associada a componentes inteligentes está a constatação de que a expansão dos computadores na moderna sociedade também alastra a estupidez computacional, com crescentes custos e riscos. Uma estupidez a ser diariamente detectada em processadores de texto, em planilhas, em programas gerenciais, em *browsers* da Web, em software educacional, etc. Para Riesbeck, a meta prioritária da IA não seria, de imediato, construir agentes inteligentes cada vez mais autônomos. Seria, antes de tudo, o desenvolvimento de *componentes inteligentes* tendo em vista fazer avançar o funcionamento dos sistemas computacionais pela redução da estupidez maquinal. Uma tal meta torna-se exeqüível, no curto prazo, por exemplo, através da busca de uma certa quantidade de técnicas básicas a exemplo das: *(i)* estruturas de *conhecimento explicito*; e *(ii)* dos recursos da *memória de casos*.

1.4.3.1 Componentes inteligentes

Componentes inteligentes, portanto, são partes integrantes de um outro sistema mais amplo. Em decorrência disto, o *design* de componentes pode ser de ordem de magnitude muitas vezes mais simples do que o *design* de agentes inteligentes complexos. Nesta visão de IA, os componentes de software se caracterizam por **[RIE 96]**:

(i) *Simplificação do design*. A simplificação ocorre devido às fortes limitações de necessidades/capacidades impostas pelo sistema mais amplo. Um componente que seja analisador sintático, por exemplo, não tem de entender, sintaticamente, toda e qualquer coisa que lhe

chegue na forma de entrada. Terá de entender, tão somente, aquelas coisas para as quais o sistema mais amplo tenha sido projetado.

(ii) *Selecionar* e *adaptar*. Componentes, fundamentalmente, têm a tarefa de ajudar sistemas maiores em tarefas de *selecionar* e *adaptar* informações tais como respostas a situações. Tais componentes podem ser desenhados para selecionar respostas já embutidas em sistemas, de modo a não acrescentar qualquer esforço extra ao resto do sistema. Respostas selecionadas podem até estar erradas, mas existirá a chance de posterior substituição por outras respostas melhores.

(iii) *Selecionar* e *adaptar* × *gerar* e *testar*. Esta propriedade dos componentes inteligentes de poderem ser selecionados e adaptados contrasta com a visão clássica de *gerar* e *testar* (em IA e nas demais áreas da Computação). Gerar ou construir soluções (via algum mecanismo tal como o refinamento de *templates*, por exemplo), em geral, força o resto do sistema a entender estas soluções, torna difícil a limitação do tempo de geração e, se esse tempo for erradamente calculado, a solução gerada poderá ficar incompleta ou mesmo não utilizável.

(iv) *Dificuldades*. O *design* de componentes também apresenta dificuldades. Entre outras, o projetista não pode controlar qual o componente a fazer uso em dado momento. E, pior ainda: componentes não podem requerer níveis de complexidade de engenharia e de manutenção incompatíveis com aquele valor particular que um certo componente venha a agregar ao sistema como um todo.

1.4.3.2 Componentes inteligentes baseados em casos (CIBC)

A tecnologia do RBC, neste contexto, se coloca a serviço desta tecnologia de componentes inteligentes. Seu novo papel, tanto no seu algoritmo geral (objeto do Capítulo 2) quanto nos seus algoritmos especializados, é o de contribuir para a modelagem dos denominados *Componentes Inteligentes Baseados em Casos* (CIBC). A contribuição do RBC nesta modelagem de componentes é relevante sobretudo em tarefas envolvendo:

(i) *Pronta seleção de respostas*. A seleção de respostas via RBC viabiliza uma propriedade útil aos componentes inteligentes que é a *prontidão* de respostas. Isto é, estes processos são capazes de encontrar uma resposta quase que imediatamente, permitindo, em seguida, a substituição de uma resposta provisória por alguma outra resposta mais apropriada.

(ii) *Adaptação de respostas*. Também o RBC viabiliza a adaptação de respostas, se necessária. Nesta hipótese, porém, adaptações só fazem sentido se forem rápidas e limitadas, para não herdarem aquelas mesmas desvantagens encontráveis nos processos envolvendo *ge-*

ração e *teste*, a serem descritos na seção 3.6.3.1. Processos de adaptação podem até mesmo serem evitados como já foi demonstrado por abordagens de RBC do tipo *Retrieval and Propose* (cf. seção 2.8.3).

Componentes inteligentes baseados em casos que sejam capazes de selecionar/adaptar respostas a partir de um estoque de respostas indexadas podem ser integrados a sistemas maiores. Na Figura 1.2 é mostrada uma forma de integração entre componentes inteligentes baseados em casos e sistemas mais amplos.

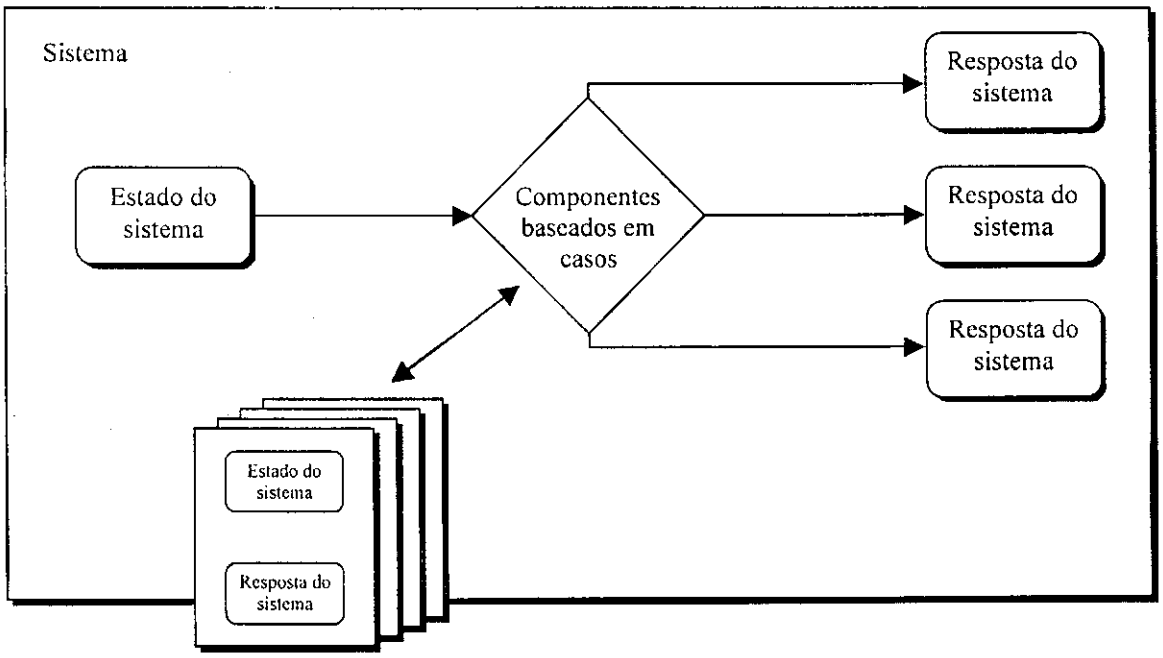

Figura 1.2: Componentes inteligentes baseados em casos [RIE 96].

Exemplos de sistemas baseados em componentes inteligentes são aqueles apontados por Riesbeck: o *Analisador Sintático Casper*, o *Criticador Casper* e o *Analisador Creanimate*, desenvolvidos no Institute for the Learning Sciences (Northwestern University, USA).

1.4.3.3 Perspectivas de evolução do RBC

Em que direção então deverão progredir as aplicações e a própria teoria dos CIBC? A este respeito, Riesbeck também analisa esta tendência dos componente inteligentes a partir da projeção dos seus empregos atuais e do estado da arte do RBC. A perspectiva é a de que os CIBC venham a evoluir em três direções principais: *(i)* bases de casos encapsuladores de <situação, resposta>; *(ii)* bases de casos navegáveis; *(iii)* bases de casos para sistemas inteligentes, propriamente.

Bases de casos encapsuladores de <situação, respostas>

Encapsular ou estocar respostas para questões freqüentemente colocadas (*Frequently Asked Questions* – FAQ) é, com certeza, uma das formas de eliminar aquela que tem sido uma das principais manisfestações de estupidez nos sistemas. Ou seja: eliminar/reduzir a estupidez de ter de preparar uma mesma resposta, cada vez que reapareça um idêntico/mesmo problema, sem considerar sequer a quantidade de vezes que este mesmo problema já tenha sido resolvido. O RBC constitui uma alternativa tecnológica fundamental para estas questões, até o presente momento, um território próprio da área computacional dos *Flexible Query Answering Systems* [**AND 97**] a se desenvolver *paralelamente* ao RBC, como ver-se-á no Capítulo 3 (cf. seção 3.4). Empregar casos para encapsular <situação, resposta>, no entanto, pode apresentar dificuldades sempre que decrescer a coincidência de uma situação passada com uma situação presente, sendo o resgate e a adaptação de casos as válvulas do RBC para resolver estes problemas.

Bases de casos navegáveis

Uma outra promissora classe de componente inteligente já em desenvolvimento são as bases de casos para efeito de navegação. O usuário utiliza o sistema para fazer *zoom* a casos de seu interesse. A partir destes casos iniciais que empregam *links* em sua concepção, o usuário pode alcançar outras variedades de casos, tal como em MEDIATOR e Protos [**KOL 93**]. Aspecto importante nesta classe de emprego do RBC é o fato de que as bases de casos navegáveis têm a vantagem de permitir ao usuário mergulhar no conteúdo dos casos computacionais e deles fazer uso sem, contudo, recorrer aos processos algoritmicos próprios do RBC convencional. Ou seja, os casos são consultados por navegação, sendo que esta utilização dispensa qualquer aplicação de mecanismos de *similaridade* e de *resgate de casos* e também qualquer aplicação de mecanismos de *adaptação de casos*.

Bases de casos para sistemas inteligentes

Finalmente, componentes inteligentes também estão a evoluir em relação à capacidade de uso combinado de paradigmas diferentes. Sempre que uma aplicação de RBC correr o risco de levar o sistema a cometer estupidez, por exemplo, gastando mais tempo em adaptações do que seria admissível, o sistema poderá recorrer ao emprego de um outro paradigma coadjuvante. São representativos desta combinação de componentes inteligentes o sistema CABARET e o sistema CASEY (detalhado no Anexo A). O primeiro combina um componente de regras com um componente baseado em casos no domínio jurídico, enquanto o segundo sistema também combina regras e casos mas no domínio da explicação médica de diagnósticos [**KOL 93**].

1.5 Conclusão

O presente capítulo tratou de posicionar a tecnologia do RBC em relação a dois aspectos básicos: *(i)* sua aplicabilidade industrial; e *(ii)* o seu lugar dentro da IA. Sobre o papel do RBC, foram analisadas duas concepções que correm em paralelo em IA: o RBC como tecnologia de agentes inteligentes e o RBC como tecnologia de componentes inteligentes. Particularmente, procurou-se examinar o papel, as vantagens e a evolução dos componentes inteligentes baseados em casos.

Capítulo 2

Processos fundamentais do RBC

2.1 Introdução

Casos constituem a estrutura básica de encapsulação de conhecimento no contexto do paradigma computacional do Raciocínio Baseado em Casos (RBC). Por outro lado, a manipulação dos casos constitui o problema central do raciocínio envolvido. Este problema central, de fato, não constitui um problema unitário. A manipulação computacional de casos é um problema tão multifacetado e complexo que se desdobra em pelo menos cinco outros subproblemas que consistem no seguinte **[MAH 95]**:

(i) *Indexação*: diz respeito à identificação daqueles atributos que um problema anterior (um caso) precisa ter para que possa ser relevante para um novo problema subseqüente;

(ii) *Resgate*: consiste, propriamente, em fazer retornar para o usuário casos já armazenados na memória. A operação é guiada, em um primeiro passo, pelas descrições/restrições de um problema novo de entrada (um passo também chamado de *apresentação*); em um segundo passo, a meta é aproximar, o mais estreitamente possível, as descrições de entrada aos casos a resgatar;

(iii) *Seleção*: consiste no processo de escolher entre vários casos resgatados tendo em vista encontrar uma solução para aquele problema de entrada. O "melhor" caso é então selecionado para adaptação;

(iv) *Modificação/Avaliação*: modificação e avaliação são subprocessos da *adaptação* de casos. Modificação de um caso selecionado é o processo de alterar partes contidas na descrição de um caso tendo em vista alcançar um alvo. Avaliação de um caso modificado consiste em checar a viabilidade de uma nova descrição de caso; e

(v) *Retenção*: uma vez uma solução tendo sido resgatada, selecionada, modificada e avaliada, finalmente, essa solução deverá ser acrescida à base de casos, prevendo-se assim um futuro emprego para esta mesma solução – um processo hoje conhecido como *aprendizagem por experiência*.

Indexação, *Resgate*, *Seleção, Modificação/Avaliação* e *Retenção* constituem apenas um dos modos de englobar essas diversas tarefas do RBC, entre outros. Como então melhor entender estes processos? Quais as tarefas envolvidas em cada um destes processos? Que problemas continuam a desafiar a tecnologia do RBC e as suas aplicações? O presente capítulo oferece uma síntese deste universo de processos e problemas do RBC dentro do qual vai se inserir toda a nossa contribuição.

Este capítulo está organizado da seguinte forma. Na seção 2.2, introduz-se o conceito de modelagem computacional de casos do RBC, seguido de exemplificação. Também introduz-se o conceito de espaço de casos. Nas seções 2.3 e 2.4, discute-se o algoritmo geral do RBC e o modo como ele costuma ser decomposto em subtarefas. Finalmente, as seções entre 2.5 e 2.10 tratam de conceituar e examinar mais detalhadamente processos tais como de: indexação, armazenagem, resgate, similaridade, adaptação, *ranking* e avaliação de casos do paradigma de RBC, tais como vistos na literatura.

2.2 *Design* de casos: conceito, conteúdo e espaço de casos

Os casos, em RBC, devem incluir dois componentes básicos: *(i)* a descrição de uma situação problemática; e *(ii)* a descrição de uma solução para esta mesma situação, tal como no conceito aqui formulado.

2.2.1 Conceito

Pode-se definir *caso* como

> [...] *uma porção de conhecimento contextualizado – um "chunk" – e que representa quer uma experiência passada sobre Problema/Solução (casos concretos ou acontecidos – "true cases") quer uma experiência hipotética, também sobre Problema/Solução (casos abstratos ou previstos para acontecer – "abstract cases").*

Ou seja **[KOL 93]**,

$$Caso = Problema + Solução$$

Também, é comum enxerga-se um caso como um objeto computacional com três componentes básicos:

$$Caso = Problema + Solução + Resultado$$

Nesta hipótese, o componente *Resultado* a integrar um caso vai consistir de uma descrição computacional sobre como uma certa *Solução* (no passado) se comportou em relação a um certo *Problema* enfrentado pelo usuário. Trata-se, por conseguinte, de um componente para fins de avaliação da

eficácia ou não de uma certa ação tomada ou de uma certa solução que tenha sido recomendada pelo sistema para um problema particular.

2.2.2 Modelagem de conteúdos de casos

Problema, *Solução* e, eventualmente, descrições sobre *Resultado* de soluções, não é tudo que possa ficar contido em casos. Ferramentas de RBC ora em desenvolvimento ou mesmo já disponíveis no mercado (tais como *CBR Express*, ESTEEM, KATE, REMIND, S_3-CASE **[ALT 95]**) permitem também incorporar aos casos que estejam sendo modelados todos aqueles tipos de dados normalmente encontráveis, por exemplo, em modernos bancos de dados. Casos computacionais poderão também incluir:

(i) *Rótulo* ou *Título*: servindo de identificação para um caso disponível em bases de casos;

(ii) *Indagações*: que, em tempo de execução, ajudam o usuário a aceitar ou não um certo caso.

(iii) *Textos em Linguagem Natural*: conforme o exemplo a ser apresentado mais à frente.

(iv) *Ponteiros*: apontando para um outro caso de interesse ou para outras fontes de conhecimento extra casos.

(v) *Multi-Mídia*: fotos, sons e imagens de vídeo também podem compor os casos ou podem a eles virem associados.

Todos estes componentes, de um modo genérico, costumam ser descritos em termos de um conjunto de pares de *<atributo, valor>* como segue **[MAH 95]**:

<Rótulo do Caso>:
<Atributo$_1$>: <Valor$_1$>
<Atributo$_2$>: <Valor$_2$>
<Atributo$_3$>: <Valor$_3$>
.
.
.
<Atributo$_n$>: <Valor$_n$>

O objeto ou situação representado pelo caso, neste exemplo, está sendo representado através daquelas suas características *ontologicamente* mais essenciais (*indexação baseada em atributos*).

2.2.2.1 Problema em *help desk*: exemplo

Exemplifica-se a seguir este processo de modelagem de casos, no domínio de *help desk*, um domínio onde os casos da base de conhecimento devem, por definição, providenciar algum tipo de ajuda bem específica para a solução de um problema, também bem específico. Em *help desk*, a ajuda em

matéria quer de *hardware* quer de *software* tem sido, particularmente,de larga aplicabilidade, como no exemplo a seguir.

Suponha, inicialmente, uma classe simples de problema que costuma acometer periféricos tais como impressoras a lazer: um problema – que no *catálogo de bugs* de um sistema aparece como sendo o problema de número 14 **[HDI 97]** – e que é identificado através da seguinte mensagem, no visor da impressora em pane: "*14 Lower Tray*". Ao modelar-se um caso para este problema (utilizando-se, por exemplo, uma ferramenta como *CRB Express*), um bom *Título* poderia ser: "*Bandeja cassette inferior está instalada inapropriadamente*". Atribuindo-se deste modo um título apropriado ao caso, o passo seguinte será a descrição do *Problema* a constar naquele caso, a ser computacionalmente armazenado. Esta descrição deverá refletir o modo como um usuário descreveria o seu problema e seus sintomas para um engenheiro de hardware. Tomando-se ainda como referência a ferramenta *CRB Express*, ela permite uma modelagem de casos, em alto nível, aceitando as descrições na própria linguagem natural do projetista. A descrição desse problema a ser embutida em um caso poderá ter o seguinte conteúdo:

Impressora não imprime nem mesmo um auto-teste. Nada acontece. Não funciona.

Imagine-se também que se queira incluir *Indagações* no *design* deste mesmo caso. Estas indagações poderão ser aquelas mesmas indagações de um engenheiro de hardware frente a frente a um usuário às voltas com essa classe de problemas.

2.2.2.2 Solução em *help desk*: exemplo

Indagações contidas em casos podem se referir a auto-teste, a possíveis mensagens que possam ser visualizadas ou podem se referir ainda a ações possivelmente já tomadas antes da chegada do atendimento de um profissional. Finalmente, a coisa mais importante a ser representada no interior de um caso será a própria solução do problema ou *ação* a ser recomendada. Em síntese, a estrutura do caso que estamos a modelar está mostrada na figura 2.1.

Rótulo do caso: Bandeja inferior está instalada inapropriamente

Descrição do problema: Impressora não imprime, nem mesmo um auto-teste. Nada acontece. Ela não funciona.

Indagações:	Respostas
Pode-se imprimir auto-testes?	Não
Mensagem mostrada no display:	"*14 Lower Tray*"
Bandeja Inferior está correta?	Não

Descrição da solução: Reinstalar a Bandeja Cassette Inferior

Figura 2.1: Caso modelado com a ferramenta *CBR Express*

Rótulo ou *título do caso*, *descrição do problema*, *indagações* e *descrição de solução* são comumente denominados de *índices* (ou *atributos* ou *termos*, mais apropriadamente) e são aqui empregados para compor este caso comum na área de suporte técnico em matéria de hardware.

2.2.3 Espaço de Casos

A Figura 2.2 sintetiza o funcionamento básico da tecnologia de casos, em termos de: (i) *espaço de problema*, e *(ii) espaço de solução* [**WAT 97**].

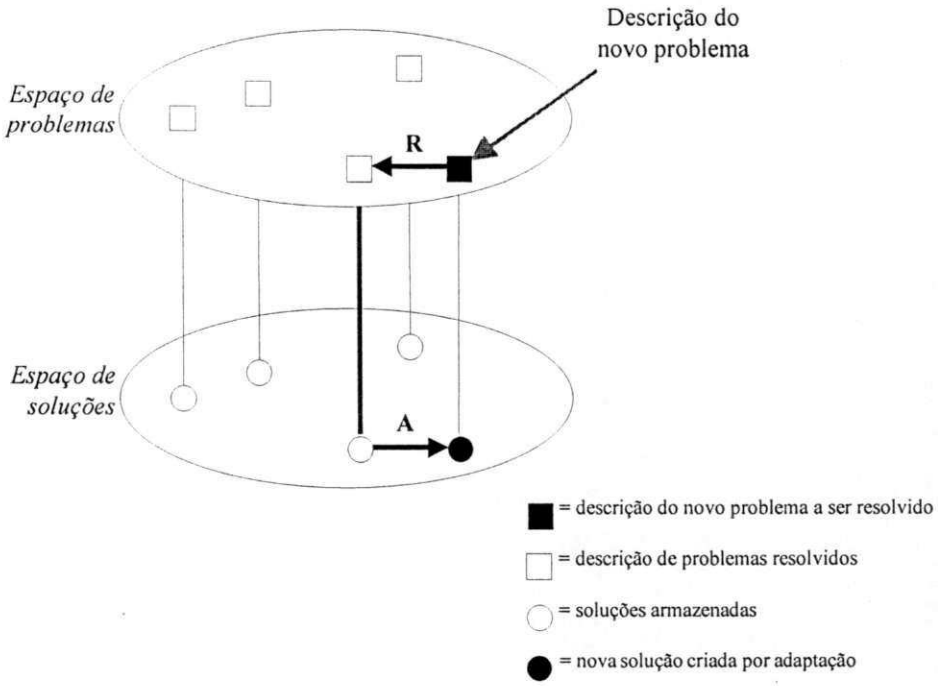

Figura 2.2: Espaços de problema e de solução

Tanto o *problema* quanto a sua *solução* giram em torno de seus respectivos *espaços*. A descrição de um novo problema a ser resolvido (quadrado em negrito) é colocada no espaço do problema (via uma operação de *apresentação*). Uma operação de *resgate* então vai identificar um certo caso como sendo a descrição de um problema mais semelhante ao problema corrente do usuário (operação **R** da Figura 2.2). Isto implica que também uma solução foi localizada. Se necessário, *adaptações* também podem ser feitas (operação **A** da Figura 2.2) e uma nova solução vai ser proposta. Portanto, este modelo conceitual de RBC assume que existe uma correlação do tipo *um-para-um* a mapear os espaços de problema e de solução. Em outras termos, sempre que um problema novo venha a surgir mais à esquerda do espaço de problemas, também uma nova solução aparecerá mais à esquerda, no espaço de soluções. A Figura 2.2 tem a vantagem de ilustrar a idéia de espaços de problema/solução e a idéia de mapeamento um-para-um. Porém, ela não chega a contemplar o conjunto daquelas operações necessárias à manipulação de casos. Tratamos, particularmente, destas

operações ao descrevermos o algoritmo de manipulação de casos, na seção 2.3 seguinte.

2.3 Algoritmo geral do RBC

Foram analisados, até o presente ponto, conceitos, conteúdos e exemplo de casos, no contexto do RBC. Examinamos, na seqüência, o que fazer e como manipular casos em uma base de casos previamente criada. Esta manipulação de casos constitui uma operação complexa que envolve processos genericamente denominados de *algoritmo geral do raciocínio baseado em casos*. Uma versão compacta deste algoritmo será:

Raciocínio baseado em casos

begin

Obtem-se as especificações de um problema novo;
Identifica-se os atributos da indexação:
    Resgata-se o conjunto de casos que se emparelhem aos atributos;
    Seleciona-se um caso;
    Repete
        Modifica o caso;
        Avalia a solução;
    até que solução seja satisfatória;

end

Neste algoritmo, os processos envolvidos, em geral, são tratados em quatro etapas básicas. Cada uma delas objetivando representar e fazer uso apropriado do conhecimento encapsulado ou a ser contido em casos residentes, conforme A. Aamodt **[AAM 96]**:

(i) *Resgate* daqueles casos armazenados que sejam os mais semelhantes ao caso do usuário;

(ii) *Reutilização* de casos para tentar resolver um problema novo;

(iii) *Avaliação/Revisão* daquela solução contida em casos prévios (se necessário for);

(iv) *Retenção* de uma nova solução, de modo a enriquecer a base de conhecimento com um novo caso.

Estas etapas constituem uma outra forma de englobar aqueles passos antes introduzidos na seção 2.1. A Figura 2.3 ilustra estes processos também conhecidos como algoritmo dos 4-R.

Na parte superior da figura, o termo *problema* está a expressar qualquer caso novo do usuário ou *caso alvo* a demandar comparações com algum caso similar. Descrições compondo esse *caso alvo* são usadas para *resgatar* um ou mais casos de entre uma coleção de *casos fontes* na memória (casos anteriores). O caso resgatado então deverá ser comparado com o caso novo tendo em vista um processo de *reutilização* que deve resultar na solução daquele problema trazido pelo usuário. Ou seja, resulta na aceitação de uma sugestão de *solução* para o caso de entrada. Esta solução sugerida

pelo sistema poderá ter uma aceitação/aproveitamento total ou apenas um aproveitamento parcial, conforme sejam as necessidades do usuário. Subseqüentemente, através de um processo de *revisão*, uma solução sugerida poderá ser testada/avaliada quando de sua aplicação ao ambiente da aplicação do sistema. Nesta fase do algoritmo, uma solução contida em casos poderá, de alguma forma, ser adaptada.

Retenção, finalmente, significa que as experiências de sucesso com os casos resgatados devem ser retidas para futura reutilização. Isto significa que o sistema é capaz de *aprender casos*, justamente, por sua capacidade de atualizar uma base de casos, conforme o conceito de aprendizagem vigente no RBC. *Aprendizagem* em RBC consiste nesta *retenção* em decorrência ora da: *(i)* inclusão de um caso novo (aprendizagem rota); ora em decorrência da *(ii)* reinclusão de casos antes já existentes mas que tenham sido de alguma forma modificados ou adaptados para solucionar um problema novo desconhecido pelo sistema.

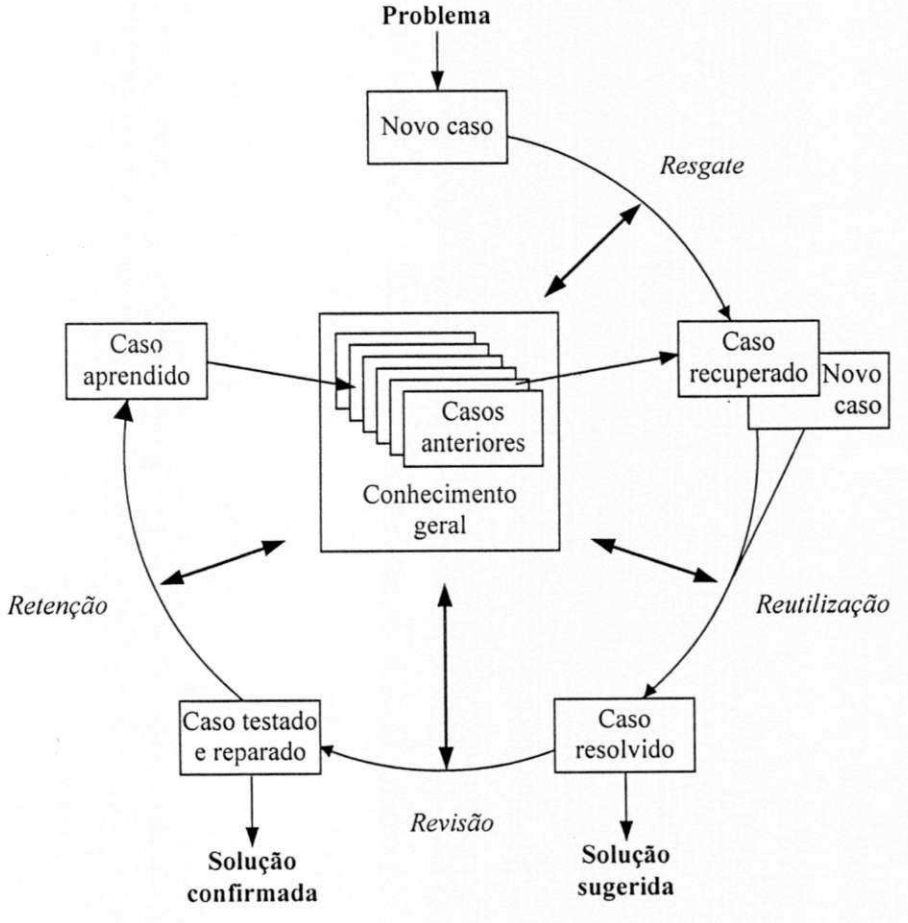

Figura 2.3: Algoritmo geral do raciocínio baseado em casos

2.4 Decomposição do algoritmo geral

Todo esse processo, assim descrito, parece simples. De fato, muitas outras tarefas do RBC estão ocultas ou não exibidas na figura 2.3 que representa apenas uma síntese dos processos envolvidos. Estas outras operações podem ser vistas dentro de uma "hierarquia de tarefas", como faz Aamodt **[AAM 91]**. A Figura 2.4 ilustra esta visão de hierarquia onde cada um dos passos, ou subprocessos, é visto como uma tarefa que o sistema de RBC precisa cobrir.

Solução de problemas e aprendizagem por experiência (RBC)	Resgatar	Identificar características	Colecionar descritores
			Interpretar o problema
			Inferir descritores
		Buscar	*Seguir índices diretos*
			Buscar na estrutura de índices
			Buscar no conhecimento geral
		Iniciar Comparações	*Calcular similaridades*
			Explicar similaridades
		Selecionar	*Usar os critérios de seleção*
			Elaborar explicações
	Reutilizar	Copiar	*Copiar a solução*
			Copiar o método de solução
		Adaptar	*Modificar a solução*
			Modificar o método de solução
	Revisar	Avaliar a solução	*Avaliação por expertos*
			Avaliação do caso real
			Avaliação do modelo
		Reparar a falha	*Auto-reparo*
			Reparo pelo usuário
	Reter	Extrair	*Descritores relevantes*
			Soluções
			Justificativas
			Métodos de solução de problemas
		Integrar	Rodar o problema novamente
			Atualizar o conhecimento geral
			Ajustar índices
		Indexar	Generalizar índices
			Definir índices

Figura 2.4: Decomposição do algoritmo geral do RBC

Tem-se uma hierarquia de tarefas organizada sob três perspectivas: *(i)* tarefas, propriamente; *(ii)* métodos; e *(iii)* modelos de conhecimento do domínio de aplicação. Na figura, letras normais (não itálicas) estão a expressar as tarefas próprias do algoritmo dos 4-R. Por outro lado, os itens escritos em *itálico* estão a expressar os métodos do RBC empregados nas tarefas. As ligações entre ambos indicam decomposições das tarefas em subtarefas e métodos. Um método, para dar conta de uma tarefa, precisa do conhecimento sobre a aplicação geral do RBC em um determinado domínio, bem como de informações a respeito do problema real e de seu contexto. A Figura 2.4 não chega, no entanto, a mostrar quaisquer estruturas de controle sobre as subtarefas envolvidas, apesar de um certo ordenamento destas subtarefas, de cima para baixo, a indicar uma ordem de execução.

2.5 Indexação e representação de casos

A tarefa fundamental, em quaisquer das aplicações do RBC, consiste em extrair do problema (do contexto ou da situação em apreço) e da solução (ou lição) um conjunto de rótulos descritores que passem a caracterizar e compor cada caso. Esta tarefa de extração de rótulos descritores é considerada como sendo a tarefa fundamental justamente porque dela depende todos os demais processos do RBC. Estes rótulos para identificar atributos de casos são por nós denominados de *termos* mas, impropria e dominantemente, são também chamados de *índices* (uma denominação a ser discutida oportunamente). Da extração destes descritores de atributos dependerá o sucesso não só da tarefa posterior de resgate de casos da memória como de toda a aplicação de RBC [WIL 97].

Indexação é a denominação dada à tarefa de extração de rótulos ou índices e a respectiva atribuição desses rótulos a casos. Definidos quais índices devem integrar um caso, a modelagem prossegue em termos de representação computacional: como então representar estes componentes (usando números, símbolos, objetos)? Que tipos de representação de conhecimentos serão necessários (vetores, regras heurística, *frames*, redes semânticas, linguagens orientadas à objetos)? A decisão, porém, sobre a escolha de algum formalismo de representação costuma depender do nível de estruturação dos casos a serem modelados em um certo domínio. Examinamos, na seqüência, este aspecto particular da estruturação de casos.

2.5.1 Representação pouco estruturada de casos

Consideremos o exemplo concreto de extração de índices para a construção de casos em um domínio real tal como o de AGÊNCIA DE VIAGEM. Um caso aqui, certamente, terá de conter *índices* ou *atributos* tais como: *nome do hotel, preço, cidade, mês*. Cada índice ou atributo pode assumir um valor de entre uma lista de valores possíveis: *nome do hotel* será um *string*; *preço* será um número entre R$50 e R$300; *cidade* será um símbolo entre {SP, RJ, Recife, Fortaleza,...}; *mês* será

um atributo ordenado entre {Jan, Fev, ..., Dez}. Se o sistema de reserva de hospedagem em apreço não comportar maiores exigências, cada um dos seus casos da base de conhecimento poderá ter esse mesmo conjunto de índices descritores, como visto na figura 2.5.

	Nome	*Preço*	*Cidade*	*Mês*
Caso 1	Mascote	50	SP	Dez
Caso 2	Glória	300	RJ	Jan

Figura 2.5: Índices e seus valores em casos pouco estruturados

Aplicações onde cada caso pode ser descrito por um mesmo conjunto de índices descritores são aplicações pouco estruturadas por conterem casos também projetados com pouca estruturação. Por definição, conceitos como "*pouco estruturado*" e "*bem estruturado*" em RBC se referem à *maior diversidade* ou à *menor diversidade de tipos de dados* que a construção dos casos venha a requerer, em um certo domínio [**ALT 95**]. Neste sentido, uma representação pouco estruturada de certo domínio pode se tornar uma representação mais estruturada se tipos diferentes de índices forem acrescidos aos casos em construção.

2.5.2 Representação bem estruturada de casos

Nos casos 1 e 2 ilustrados na Figura 2.5, outros índices podem ser introduzidos de modo a mudar a estruturação dos casos. Podem ser introduzidos outros *símbolos ordenados*, *números reais*, *disjunções* (por exemplo, "*mês é janeiro ou fevereiro*") e até mesmo *regras* controladoras da abrangência de certos atributos. Conseqüentemente, diz-se que um domínio dá origem a representações bem estruturadas se existirem casos com diferentes classes de atributos.

No mesmo domínio de AGÊNCIA DE VIAGEM, se os casos tiverem de representar a hospedagem de clientes quer optando por *quarto de hotel* quer optando por *chalé*, certamente isto já implica que o sistema terá uma representação de casos muito mais estruturada. Será uma representação mais estruturada na medida em que terão de ser acrescidos outros descritores para o objeto *quarto de hotel* de modo a diferenciar este objeto de um outro como o *chalé*. Nesta hipótese, um atributo tal como *conforto 5 estrelas* poderá ser selecionado para o objeto *quarto de hotel*, enquanto que um outro atributo diferente, tal como *cozinha*, deverá ser um índice a ser acrescido para caracterizar e diferenciar o objeto *chalé*.

2.5.3 Indexação seguindo diretrizes

A ação de criar/extrair índices apropriados a casos requer uma profunda *análise de situações* que

permita dela extrair aqueles elementos essenciais a serem computacionalmente representados (*situation assessment*). Existem diretrizes para a operação de definir índices a comporem possíveis casos. Entre outras diretrizes, estão as que estabelecem que os índices devem **[WIL 97, KOL 91, KOL 93, KOL 96]**:

- servir para predizer o futuro emprego daqueles casos que venham a contê-los;
- servir aos fins para os quais servirão os próprios casos;
- ser suficientemente abstratos para servirem a usos futuros da base de casos; mas também
- ser suficientemente concretos para serem reconhecidos no futuro.

Suponha a escolha de índices *para* a construção de casos – por exemplo – no domínio de análise de *empréstimo bancário*. Casos de empréstimos passados deverão ser posteriormente usados para embasar decisões sobre a concessão ou não de novos empréstimos a clientes de bancos (como será visto no Capítulo 7). *Índices* apropriados, neste contexto, são aqueles que dão aos casos o poder de prognosticar ocorrências relativas a empréstimos bancários a clientes.

Seguindo essas diretrizes, por exemplo, índices tais como *sobrenome* do tomador de empréstimo, *número do telefone*, *CPF* em quase nada concorrerão para ajudar o analista de crédito a prognosticar ou "prever" se um certo empréstimo a ser negociado com este cliente tem condições de ser, por exemplo, amortizado sem maiores problemas. Outros índices, ao contrário, têm o poder de fornecer previsões de que o referido empréstimo possa ser proveitoso tanto para emprestador quanto para tomador. Índices tais como *endereço* (em bairros bem localizados), *nível salarial*, *nível de endividamento* (ou comprometimento financeiro atual), etc, são índices com esta propriedade. Durante uma análise de situação (uma operação de *loan-underwriting*, no exemplo), tanto pode ser escolhido o *salário* quanto o *endividamento* financeiro de um cliente de banco como os índices mais prognosticadores e, por conseguinte, os de maior interesse para os objetivos de uma base de casos, no domínio de empréstimos bancários.

2.5.4 Métodos de indexação

A literatura normalmente faz a citação do que chama de *métodos de indexação*, quando, na maioria das vezes, se trata do emprego de diretrizes (*guidelines*). A lista de tais diretrizes/métodos apontados por Ian Watson compreende **[WAT 97, p. 22]**:

- *Indexação por atributos e dimensões*. Consiste na aplicação de diretrizes para a seleção de atributos mais específicos e atributos mais genéricos ou *dimensões*. O sistema MEDIATOR **[KOL 93]**, que trata de resolver disputas entre pessoas e países que invadem a pro-

priedade um do outro, é citado por Watson como um exemplo de indexação por atributo. Os criadores do sistema procuram indexar segundo o tipo e as funções dos objetos em disputas; como também segundo os relacionamentos entre os disputantes.

- *Indexação baseada em diferenças*. Consiste em buscar e fazer uso das diferenças entre situações. Nesta hipótese, os índices seriam aquelas características que diferenciam um objeto (um caso já na memória) de um outro que está sendo indexado, como nos sistemas CASEY e CYRUS **[KOL 93]**.

- *Indexação baseada em similaridades/explicação*. Métodos de generalização baseados em similaridades e explicação, segundo Watson, podem levar à produção de um conjunto de índices para casos abstratos ou índices para casos com características compartilhadas. São métodos que usam a aprendizagem baseada em explicação (*explanation-based learning*) para extrair uma combinação de características (atributos) que levam em conta a generalização de uma explicação.

- *Indexação por aprendizagem indutiva*. Os métodos de aprendizagem indutiva também são aqui citados como métodos de indexação, utilizados para identificar e selecionar índices que funcionem como características prognosticadoras para casos do paradigma RBC (cf. seção 2.7.6 e seção 7.6.4).

2.6 Armazenagem de novos casos

Supondo-se que os casos já tenham sido *projetados*, como então organizá-los na memória para resgatá-los oportunamente? Quer estejam eles pouco ou muito estruturados, os casos precisam ficar organizados na memória de um modo adequado à tarefa do resgate. Comumente, um caso é individualmente tratado na forma de uma representação unitária, como sendo um objeto resgatável por inteiro e utilizável no todo ou em parte.

Algoritmo *acrescenta-casos* à memória

Para acrescentar-se um novo caso projetado a uma base de casos, o algoritmo exemplificado abaixo faz uso de uma lista de indexação que associa <*lista-nomes-casos*, *lista-pares-atributo*>:

Acrescenta caso

begin

Acrescenta o rótulo do caso tomado no índice dos casos;
Para cada par <$atributo_i$, $valor_i$> constando no novo caso
    Obter a lista de rótulos de casos associada com o $atributo_i$ presente na lista de indexação;
Se a lista de rótulo de casos estiver vazia
então
    Acrescenta o $atributo_i$ na lista de indexação;

```
    Cria a lista contendo o rótulo do caso;
    Associa esta nova lista de rótulos de casos ao atributo_i;
senão
    Acrescenta o rótulo do caso na lista de rótulos de casos;
end
```

Para cada par <*atributo*$_i$, *valor*$_i$> de um novo caso projetado, uma lista de indexação é pesquisada tendo em vista achar uma lista de rótulos de casos associados com o *atributo*$_i$. Um novo rótulo de caso será então acrescido à base se o *atributo*$_i$ já existir na lista porém associado a rótulo diferente; senão, esse *atributo*$_i$ será acrescido à lista de indexação.

2.6.1 Armazenamento sequencial: *flat library*

Em relação aos métodos de armazenagem destes casos assim criados, o projetista, inicialmente, precisa fazer um balanço entre métodos que venham a preservar a riqueza dos casos e de seus índices e os métodos que valorizam a simplificação do acesso e do resgate dos casos.

Também um balanço é necessário entre métodos de interesse mais acadêmicos (tal como o método da *memória dinâmica* de Schank e Kolodner [**KOL 93**]) e os métodos de interesse mais industrial que privilegiam a armazenagem de casos como estruturas de dados seqüenciais. É o chamado método de armazenamento em *flat files*. Casos ainda têm sido armazenados de modo similar ao armazenamento requerido em *bases de dados relacionais* convencionais ou ainda armazenados nas formas de *árvores de discriminação/redes compartilhadas*, que examinamos na seqüência.

2.6.2 Armazenamento hierárquico: árvores de discriminação

Bases de casos contendo uma vasta quantidade de casos requerem que estes casos sejam organizados na memória de uma forma hierárquica. A hierarquia de casos tem a vantagem de permitir que, em um dado momento, somente um subconjunto daquele conjunto de casos em memória possa ser considerado para a tarefa de resgate. Duas são as formas mais comuns de *armazenagem hierárquica* de casos [**KOL 93**, p. 300]:

- Armazenamento em redes de características compartilhadas (*Shared-Feature Network*)
- Armazenamento em redes ou árvores de discriminação (*Discrimination Network*)

A diferença entre ambas essas formas de armazenagem de casos reduz-se a uma questão de ênfase: tanto o método das redes de características compartilhadas quanto o método das árvores de discriminação são capazes de gerar agrupamentos (*clusters*) de casos similares ao tempo em que também fazem a *discriminação* entre estes casos em relação a um certo atributo previamente definido. No

primeiro método, porém, o *clustering* é o aspecto fundamental; enquanto a *discriminação* é o aspecto secundário. No método de armazenamento em árvores tem-se o resultado inverso.

Em uma árvore de discriminação, cada nó interior representa um lugar na árvore onde uma possível indagação possa ser feita para checar a presença ou não de um caso que satisfaça a essa indagação. Esta indagação serve então para discriminar ou dividir em sub-nós os itens abaixo dela. Suponha que os vários casos organizados em estrutura de árvore estejam a modelar diferentes tipos de disputas humanas, por exemplo. Neste domínio, então, um nó interior da rede poderá corresponder a uma situação de quem esteja à busca de casos para satisfazer à indagação: *qual o tipo de disputa*? A partir deste nó, descendo então na rede, o subnó$_1$ poderia alojar um caso de *disputa física* enquanto o subnó$_2$ poderia dar conta de um caso de *disputa política*.

2.7 Resgate de casos: métodos usuais

Operações de resgate de casos têm início quando, a partir do "*pronto*" de um sistema de resgate – o *memory prompt* – tiver sido dada entrada a um conjunto de restrições/condições a serem satisfeitas quando o sistema tentar retornar um ou mais casos que sejam *válidos* para um novo problema do usuário. Esse novo problema do usuário também é descrito em termos de seus atributos, como no caso abaixo.

<Novo Problema>:
<Atributo$_1$>: <Valor$_1$>
<Atributo$_2$>: <Valor$_2$>
<Atributo$_3$>: <Valor$_3$>
.
.
.
<Atributo$_k$>: <Valor$_k$>

Se cada uma destas restrições de entrada for considerada independentemente, cada restrição poderá ser usada para selecionar subconjuntos de casos na memória. Se, porém, n condições forem lançadas na posição de pronto do sistema então n subconjuntos de casos podem ser resgatados como o resultado de acoplamentos ou *matching* na memória. A interseção destes subconjuntos vai resultar naquele correspondente conjunto válido de casos obtido através do resgate total.

2.7.1 Conceito de resgate

Define-se então o pronto de um sistema de resgate de casos como sendo a tupla:

$$pronto = \langle S, Q, R \rangle$$

onde $S = s_1, ..., s_k$ representa fatos iniciais conhecidos sobre casos a buscar na memória, $Q = q_1, ..., q_m$ representa solicitações adicionais de informação e $R = r_1, .., r_n$ representa restrições sobre o atri-

butos de entrada para o resgate.

Resgate *válido* será aquele capaz de atribuir valores para todos os elementos de Q para os quais valem as restrições em R. Na prática, as propriedades deste resgate vão depender do algoritmo particular empregado.

O algoritmo *resgata-casos*, na seqüência, resgata aqueles casos da base cuja lista de pares <*atributo, valor*> se emparelha à descrição do novo problema.

Resgata caso

begin

Para cada par <$atributo_j$, $valor_j$> no novo problema
    Encontrar a lista de rótulos de casos associada com o $atributo_j$ da lista de indexação;
    Se a lista de rótulos de casos for não vazia
    então
        para cada $caso_i$ da lista de rótulos de casos
        obter o $valor_i$ do $atributo_j$ no $caso_i$;
    Se $valor_j = valor_i$
    então
        Se $caso_i$ não estiver em *casos-resgatados*
        então
            acrescentar o $caso_i$ em *casos-resgatados*;

end

Portanto, para cada $atributo_j$ em um novo problema, se os valores desse $atributo_j$ em algum dos casos e no novo problema se emparelharem esse caso será então resgatado e armazenado numa variável chamada *casos-resgatados.*

Analisamos, na seqüência, os seguintes métodos de resgate de casos: *(i)* resgate por emparelhamento de padrão; *(ii)* resgate baseado em índices; *(iii)* resgate via algoritmos seqüenciais; *(iv)* resgate via algoritmos paralelos; *(v)* resgate via algoritmos indutivos; *(vi)* resgate baseado em bancos de dados; *(vii)* resgate baseado em memória; *(viii)* resgate baseado em similaridades.

2.7.2 Resgate por emparelhamento de padrão (*Pattern Matching Retrieval*)

Um dos principais mecanismos de resgate (não apenas no RBC como em outras partes da computação) envolve a seleção de dados baseada numa *função* simples de emparelhamento de caracteres (*strings*). Trata-se de um método de busca seqüencial (nos termos da seção 2.7.4), baseada em *strings* e tem sido um método muito flexível. Quase toda combinação imaginável de caracteres pode ser emparelhada a uma velocidade razoável. Contudo, buscas baseadas em *string* costumam não oferecer muita orientação ao usuário. O método pode levar à especificações de entradas ou "prontos" inapropriados. Mesmo dando-se entrada a "prontos" apropriados, isto é, mesmo quando *strings* de entrada estejam corretamente especificados visando-se obter emparelhamentos exatos

(*exact matching*), o método não funciona de modo suficientemente robusto. Muitos resgates espúrios ou indevidos de casos vão inevitavelmente acompanhar aqueles resgates válidos. Duas dificuldades básicas dominam esse método de resgate de casos: *(i)* sua incapacidade intrínseca de, em um mesmo resgate, associar descrições sintáticas e semânticas; *(ii)* o fato de requerer a interseção de resultados parciais (ou conjuntos de casos) obtidos em diferentes buscas. Além deste particionamento de resultado por cada aplicação do método, cada conjunto vem acompanhado de casos espúrios, sendo somente poucos casos relevantes.

2.7.3 Resgate baseado em índices (*Indexing-Based Retrieval*)

O emprego de índices costuma ser um método de resgate de casos (impropriamente também denominado de *indexação*) mais adequado do que o simples emparelhamento de *string* porque ele tem a vantagem de reduzir os custo da busca de casos **[BRO 95]**. Os índices, portanto, têm o papel de permitirem o acesso aos casos de um modo muito mais eficiente, uma vez que o custo da busca costuma ser menor do que no método anterior de emparelhamento de padrão. Indexação, neste sentido da literatura de RBC, requer que os atributos ou índices dos casos pré-determinem o funcionamento do resgate. Esta característica de pré-determinação significa que os caminhos de acesso a casos, de certo modo, já são rigidamente definidos e pré-determinados quando esses índices são criados/extraídos para integrar casos (*indexação*, num primeiro sentido) e quando eles são disponibilizados para o resgate via "pronto" (*indexação*, no sentido impróprio). Se, porém, a extração desse conjunto de índices for viabilizada com facilidade, se isto resultar no mais completo conjunto de índices descritores de situações e se também estes índices não vierem a sofrer constantes mudanças, ao longo do tempo, então esse emprego de índices pode se tornar um dos métodos mais apropriados de resgate. Contrariamente, se estes índices extraídos – visando tanto a composição de casos quanto ao seu posterior resgate – tiverem de mudar, quer por acréscimos quer por "deleção", toda a representação dos casos na memória, como também os caminhos de acesso a eles, terão de ser atualizados. Este fato desfavorece o método de resgate baseado em índice.

Resgate baseado em índices também é, por natureza, um método seqüencial. Uma busca empregando esse método normalmente envolve técnicas de percorrimento em árvore. Contudo, isto não exclui a possibilidade de que esse método de resgate por índices também possa funcionar como um modelo paralelo de busca. Nesta hipótese, os ramos de uma árvore também podem ser percorridos em paralelo e, por conseguinte, cada índice pode guiar uma busca paralela ao longo dos ramos de uma árvore armazenadora de casos.

2.7.4 Resgate via algoritmos seqüenciais

Casos armazenados seqüencialmente em formas simples tais como listas, vetores ou arquivos, dão origem à formas também mais simples de resgate através dos denominados algoritmos seqüenciais. Um primeiro exemplo desta classe de algoritmo já foi analisado na seção 2.7.1. Duas outras versões deste mesmo algoritmo estão mostradas abaixo [**KOL 93**, p. 294].

Resgate seqüencial

begin

Para cada caso na memória,
    Faça o(s) emparelhamento(s) com o caso alvo;
Retorne o(s) melhor(es) emparelhamento(s) de casos;

end

Resgate seqüencial modificado

begin

Se mais da metade dos atributos do $Caso_1$ são iguais aos atributos do $Caso_2$
então
    conclua ser bom o grau de emparelhamento entre ambos
senão
    imprimir "Houve fraco emparelhamento entre os casos"

end

Na primeira versão, tem-se um algoritmo simples de resgate seqüencial; na segunda versão, tem-se um algoritmo seqüencial modificado. Tanto a primeira quanto a segunda versão são algoritmos que têm recebido ressalvas por não distinguirem entre aspectos quantitativos e aspectos qualitativos dos índices ou atributos durante a tarefa de comparação de casos. A crítica que costuma ser feita é a de que um certo atributo pode possuir uma certa importância, no contexto de um caso, enquanto que, em um outro caso, este mesmo atributo poderá ter uma diferente importância e esta diferença de importância não está sendo considerada nos algoritmos acima [**KOL 91**, p. 15].

2.7.5 Resgate via algoritmos paralelos

Os algoritmos paralelos apresentam um maior poder de resgate de casos; um poder que decorre principalmente do fato de que eles visitam todos os casos (ou muitos deles) de uma só vez [**KOL 91**]. A vantagem de um massivo paralelismo pode ser identificada como sendo a sua habilidade de encontrar a interseção de um conjunto de casos, de uma só vez. Considere-se o seguinte exemplo, encontrado na literatura [**BRO 95**]:

"João queria cometer suicídio. João pegou uma corda"

A maior parte das pessoas prontamente costuma fazer a inferência de que *João* pretende usar a

corda como um instrumento de enforcamento. Porém, para se poder fazer esta inferência em RBC dispondo-se de uma representação hierárquica em rede/árvore será necessária uma busca indireta que pode ser implementada usando-se, por exemplo, *passagem de marcadores*. A técnica de passagem de marcadores (*marker passing*) é uma técnica para a realização de buscas ao longo de uma representação de conhecimento baseada em redes. No exemplo em discussão, o *resgate paralelo* seria conseguido através do uso de *passagem de marcadores*, da seguinte maneira:

1. Coloque um marcador "1" naquele nó da rede de nome "corda".
2. Encontre todos os possíveis usos de "corda".
3. Coloque um marcador "2" naquele nó da rede de nome "suicídio".
4. Encontre todos os métodos de "suicídio".
5. Extraia então quaisquer dos nós que tenham recebido ambos os marcadores (por exemplo, "enforcamento").

Em uma implementação de paralelismo massivo, as etapas 2 e 4 serão realizadas de uma só vez. Contrariamente, numa implementação seqüencial, estas duas etapas tomariam um tempo proporcional ao tamanho daqueles dois conjuntos de *nós* que o sistema tem de visitar ou acessar. Problemas neste tipo de implementação aparecem quando é alta a quantidade de soluções candidatas, ou ainda em situações onde a busca por interseções (de conjuntos de *nós*) envolve uma longa busca indireta, isto é, um resgate do tipo *não-determinístico*. Nestas situações, o emprego isolado de passagem de marcadores para viabilizar o paralelismo massivo se torna insuficiente para selecionar os melhores itens de conhecimento armazenados na memória.

2.7.6 Resgate via algoritmos indutivos

Na seção 2.6.2, foram discutidos métodos de armazenamento de casos na forma de hierarquia de casos. Casos armazenados hierarquicamente, quer na forma de *árvores de discriminação* quer na forma de *redes de características compartilhadas*, costumam ser resgatados através do emprego de algoritmos baseados em *indução computacional* ou algoritmos indutivos [**MON 95, WAT 97**, p. 28].

De um modo geral, a indução computacional constitui uma técnica desenvolvida na área de *aprendizagem de máquina* com o objetivo de gerar regras generalizadoras ou de construir árvores de decisão empregando dados passados. Em sua aplicação particular ao RBC, a indução originária da aprendizagem de máquina requer que esses dados prévios passem a corresponder aos casos a serem objeto de manipulação e raciocínio. Intuitivamente, a idéia de indução aplicada ao RBC consiste no seguinte [**KOL 93**, p. 296]:

> "Se você pode colocar juntos aqueles casos que são similares uns aos outros, de modo a ser capaz de distinguir quais grupamentos de casos melhor se emparelhem com a situação representada em um novo caso alvo, então somente aqueles itens – em um certo grupamento particular – precisam ser considerados quando da operação de busca dos melhores emparelhamentos de casos".

Nesta classe de resgate de casos, a tarefa de busca começa quando se tem um novo caso em mãos do usuário e termina quando é encontrado na árvore de discriminação um caso que melhor se acople ao caso de entrada (*caso alvo*). Essa busca começa no topo da árvore e continua até que um emparelhamento de casos seja encontrado. Deste modo, grandes porções da memória de casos passam a ser eliminadas da pesquisa desnecessária quase que imediatamente. Em uma de suas formas de viabilização, a busca de um caso armazenado em uma árvore de discriminação pode ser efetuada conforme o algoritmo de Kolodner **[KOL 93]**, apresentado abaixo.

Busca em árvore de discriminação

begin

Faça N = raiz da árvore;
Repita até que N seja um caso:
    Em N faça uma indagação relativa ao caso de entrada;
    Faça N = subnó detentor de melhor resposta à indagação feita;
Retorne N;

end

O algoritmo de indução mais largamente empregado em RBC é o ID3 **[ALT 95, WAT 97, MON 95]**, a ser exemplificado no Capítulo 7 ao compararmos nossos experimentos com a indução em ID3.

2.7.7 Resgate baseado em banco de dados (*Database-Driven Retrieval*)

Uma outra classe de resgate de casos recebe a influência direta das metodologias de banco de dados. O modo como funcionam os bancos de dados serve de base não apenas para o funcionamento das ferramentas do RBC (*shells*) como também para a sua infra-estrutura de memória e as próprias operações de resgate. De fato, modernos bancos de dados permitem que complexas indagações sejam emparelhadas aos conteúdos destes bancos e deste modo possam assim ser respondidas. Tais indagações podem até mesmo lidar com emparelhamentos ou *matching* alternativos como também lidar com o emprego de "*coringas*" (que são símbolos vazios incluídos para alargar o horizonte das indagações). Elas têm a capacidade de resgatar da memória todos os dados que venham a se emparelhar exatamente com uma descrição contida no "pronto" do sistema.

Bancos de dados, porém, padecem da rigidez das linguagens de consulta, sobretudo em relação a usuários não especialistas, e também padecem da *falta de robustez*, em relação a ruídos e a omissões de informação no "pronto" inicial do sistema. Trabalho teórico recente propõe tratar base de

casos como se fosse banco de dados, implantando métricas de similaridades no topo de banco de dados. Esta estratégia porém tem sido problemática [**BRO**, p.46]. Uma das vantagens do resgate em bancos de dados sobre o resgate em RBC baseado em índices está na habilidade de se poder construir em bancos de dados padrões de resgate de dados muito complexos. Os bancos de dados permitem, por exemplo, realizar resgates altamente indiretos e que não são apropriados ao resgate baseado somente em índices. A depender da situação, porém, pode constituir uma vantagem do resgate via índices o fato de se poder evitar em RBC o problema da busca por caminhos indiretos ou tortuosos, uma vez que o emprego de índices requer caminhos de busca diretos.

2.7.8 Resgate baseado em memória (*Memory-Driven Retrieval*)

Resgate orientado pela memória refere-se ao tipo de resgate onde aquelas restrições norteadoras da operação já não têm de ficar completamente definidas no "pronto" do sistema. Parte do controle do resgate vai depender da descoberta de índices adicionais necessários e da descoberta de caminhos indiretos para esse resgate. Isso exige que a representação/codificação dos casos e os *links* entre atributos de casos possam ser armazenados na memória de uma forma que possam ser apropriadamente visitados. Suponha a descrição de um caso onde não haja uma *direta representação* de fatos, por exemplo, fatos sobre:

Por que João Pereira estava infeliz?

Pode ser que existam várias razões para a insatisfação de João e na memória venha a ser encontrado um caminho de inferências equivalente ao que se segue:

Ao ser preso pode-se ir à condenação.
Condenação leva a punição.
Punição pode levar as pessoas a se sentirem infelizes.

Encontrar estas inferências na memória implicará, certamente, em uma busca indireta ao longo de uma rede de conhecimento sobre o mundo real e, nesta hipótese, a simples enumeração de restrições/condições no "pronto" do sistema não se adequa bem para definir completamente o caminho requerido para esse resgate. Em memórias de grande escala, o espaço de busca a navegar costuma ser gigantesco. Heurísticas potentes se tornarão necessárias, tanto para tornar o espaço de busca mais tratável quanto para filtrar aqueles casos a serem resgatados, de tal modo que os casos relevantes não dêem lugar a um grande número de casos irrelevantes.

2.7.9 Resgate baseado em similaridades (*Similarity-Based Retrieval*)

Resgate baseado em similaridades pode ser considerado como uma extensão do resgate baseado em índices, da seção 2.7.3. Seu princípio básico de funcionamento consiste em se usar *métricas de*

similaridade que possam ser aplicáveis a casos na memória de tal modo que aqueles casos a obterem os mais altos escores sejam selecionados pela operação de resgate. O problema principal aqui consiste em se saber o que vem a constituir uma apropriada métrica de similaridade.

Concepções de similaridade podem variar de uma visão simplista de similaridade como *distância geométrica* entre casos até uma concepção mais complexa apoiada em paradigmas cognitivos, conforme a crítica que empreendemos ao analisar-se o estado da arte da similaridade de casos no Capítulo 4.

O resgate baseado em similaridade possui uma propriedade interessante. A mensuração da similaridade permite alguma sorte de tolerância limitada em relação a ruídos no "pronto" do sistema. Isto ocorre ao ser possível fazer-se um certo emparelhamento entre aquelas restrições especificadas no "pronto" e apenas um sub-conjunto daqueles índices constantes em um caso armazenado. Ou seja: para casos serem resgatados não se faz necessário que *todas* aquelas restrições especificadas no "pronto" sejam satisfeitas. Neste sentido, o resgate via algoritmos de similaridades implementa uma forma de resgate por emparelhamento ou *matching parcial* que em muito se assemelha a um procedimento *fuzzy*.

Também o resgate baseado em similaridade muito claramente apoia o acesso paralelo a casos. Isto ocorre porque nada impede que uma mesma métrica de similaridade possa ser *simultaneamente* aplicada a todos os casos que estejam na memória. Esta característica tem sido defendida por muitos pesquisadores como um mecanismo rápido e também flexível de acesso a casos **[COO 91]**.

2.7.9.1 Propriedades da similaridade de casos

Muitas têm sido as medidas de similaridades propostas para selecionar aqueles casos mais relevantes. Quaisquer uma delas, no entanto, costumam exibir as seguintes propriedades **[ALT 95]**:

- *Reflexividade:* um caso é sempre similar a si mesmo.
- *Simetria*: se o caso **A** é similar ao caso **B**, então o caso **B** é sempre similar ao caso **A**.
- *Não transitividade*: se o caso **A** é similar ao caso **B** e **B** é similar ao caso **C**, *nem sempre* se pode garantir que o caso **A** é similar ao caso **C**. Um BMW branco, por exemplo, é similar a um Renault branco e um Renault branco é similar a um Renault vermelho *mas* um BMW branco *não* é similar a um Renault vermelho.

2.7.9.2 Similaridade local × similaridade global

Existem duas classes de similaridade quanto à sua abrangência: a *similaridade global* e a *similaridade local*. Tipicamente, a similaridade global entre o caso alvo (Q) e um caso fonte (C), ambos descritos por n atributos, e denotada por $SIM(Q,C)$, está baseada na computação das similaridades locais *sim* entre cada atributo destes dois casos e pode ser expressa por:

$$SIM(Q, C) = f(sim_1(q_1,c_1), sim_2(q_2,c_2), ..., sim_n(q_n,c_n))$$

com f: $[0,1]^n \rightarrow [0,1]$.

Uma classe particular de métrica para computar essa similaridade pode ser obtida pela soma ponderada das *similaridades locais* para cada atributo conforme a expressão que segue **[LEA 96]**:

$$SIM(Q,C) = \frac{\sum_{i=1}^{n} W_i \times sim(Q_i, C_i)}{\sum_{i=1}^{n} W_i}$$

onde i designa qualquer atributo particular de *1* até n; W significa um peso medidor da importância do atributo i; n significa a quantidade total de atributos em cada caso; $sim(Q_i, C_i)$ designa a similaridade particular ou local entre o valor do *i-ésimo* atributo do caso alvo (Q) e do caso (C) na base de casos.

Uma família inteira de métricas tais como a métrica *Weighted Block-City*, a *métrica euclidiana*, a *métrica de Minkowski* e a *métrica ponderada de Minkowski* funciona de modo similar ao mostrado acima (Ver Anexo A sobre as métricas de similaridade mais empregadas em RBC) .

2.8 Adaptação de casos

Um outro problema difícil em RBC é a adaptação de casos. Ao resgatar um caso da base de casos, o sistema tentará, na etapa seguinte, reutilizar aquela solução sugerida pelo(s) caso(s) recém-resgatado(s). A situação ideal pode ocorrer quando a solução contida em um caso resgatado venha a ser suficiente para resolver uma nova situação problemática. Entretanto, em outras situações, a solução resgatada poderá ficar apenas próxima da solução requerida pelo novo problema, mas não ainda *suficientemente similar* à solução necessária. A tecnologia de RBC dispõe de três alternativas que podem ser seguidas nestas situações **[KOL 93]**:

- Adaptação automática de casos
- Adaptação manual de casos

- Não adaptação de casos

2.8.1 Adaptação automática × adaptação manual

Assim como na operação de indexação, a adaptação de casos tanto pode ser uma operação realizada manualmente pelo usuário quanto uma operação automática através de regras, heurísticas, fórmulas e algoritmos. Adaptar casos, muitas vezes, corresponde a uma busca por diferenças entre o caso novo e o caso resgatado. São estas diferenças possíveis de serem identificadas que poderão ser usadas para nortear a obtenção de uma sugestão aceitável como solução para o problema em apreço.

O algoritmo que segue é capaz de, automaticamente, adaptar casos na condição de que sejam dados um certo caso a adaptar e também uma lista de pares desejados de <*atributo*, *valor*>.

Adapta caso

begin
    Para cada par <$atributo_i$, $valor_i$> em um novo problema
        Se $atributo_i$ estiver na descrição contida no caso a adaptar
        então
            Obtenha o $valor_j$ do $atributo_i$ do caso resgatado descrito;
            Se not($valor_j = valor_i$)
            então
                Substitua $valor_j$ pelo $valor_i$;
        Se $atributo_i$ não estiver na descrição do caso dado
        então
            Acrescentar o par <$atributo_i$, $valor_i$> à descrição do caso;

end

Portanto, para cada par <$atributo_i$, $valor_i$> da descrição de um novo problema (*caso alvo*), se o $atributo_i$ não vier a ser encontrado no caso dado, esse par <$atributo_i$, $valor_i$> vai ser acrescido ao caso dado. Se, o $atributo_i$ for encontrado, porém, com um valor não desejado então este seu valor vai ser substituido pelo $valor_i$.

2.8.2 Métodos de adaptação de casos

Nove métodos básicos têm sido empregados em adaptação, estando eles agrupados nas categorias de *(i)* métodos de substituição e *(ii)* métodos de transformação **[KOL 91]**. Métodos usuais de adaptação de casos são os seguintes:

- *Métodos de substituição*. São usados para – em uma solução passada – substituir um objeto, um valor, conjunto de objetos ou valores por um ou mais conjuntos que melhor se adeque a uma situação nova. São classes de substituições as seguintes:

(i) *Reinstanciação.* Significa instanciar a representação da solução passada com outros novos argumentos ou valores.

(ii) *Ajustamento de parâmetros.* Consiste em se ajustar os parâmetros de uma solução contida em um caso passado baseado nas diferenças entre as descrições dos casos novo e passado.

(iii) *Busca local.* É uma busca ao longo de uma certa hierarquia semântica objetivando a substituição de algum objeto de uma solução passada que se adeqüe a substituições.

(iv) *Memória de indagação.* É uma busca mais ampla por algum tipo de item substituível.

(v) *Busca especializada.* Ela direciona a busca para certas porções de uma base de casos onde substituições sejam mais prováveis de ocorrer.

- *Métodos de transformação.* Eles objetivam transformar, de algum modo, porções de uma solução passada para se ajustar a uma nova situação, destacando-se:

  (vi) *Transformações de senso comum.* Fazem uso de conhecimentos do senso comum sobre aquelas coisa que podem ser objeto de transformações.

  (vii) *Reparos guiados por modelos.* Emprega modelos qualitativos para guiar transformações a serem operadas em casos.

- *Aplicação crítica.* Consiste em implementar quaisquer dos tipos de adaptação acima apontados, através do emprego de heurísticas *ad hoc*, sobretudo heurísticas de inserção, deleção e reordenamentos de itens.

- *Replicação derivacional.* Não tenta adaptar a solução contida em casos nela mesma, mas sim replicar o mesmo método empregado na construção desse caso passado. É também chamada de *adaptação derivacional.*

2.8.3 Não adaptação de casos: RBC especializado

Técnicas de adaptação de casos têm provado funcionarem com sucesso em domínios bem conhecidos, bem compreendidos e freqüentemente bem comportados. A complexidade destas operações de adaptação, a dificuldade em generalizar heurísticas aplicáveis a domínios diferentes, como ainda os parcos resultados obtidos até agora, em muitos domínios, levaram ao aparecimento de uma *especialização* do RBC, simplificadora do ciclo dos 4-R (seção 2.3). É a denominada abordagem de RBC do tipo *resgata e propõe* (*"retrieve and propose"*) [**MAK 96**, p. 281, **KOL 93**, p. 60] e que corres-

ponde (em outros domínios da IA) a uma das estratégias de agentes inteligentes para implementação física de bases de conhecimento conhecida como *armazena e apanha* ("*store and fetch*"), conforme já descrito por Russell & Norvig [**RUS 95**, p. 299].

O tratamento de casos do tipo *resgata e propõe* desestimula, por conseguinte, o emprego de adapatações ou transformações automáticas que, em situações diversas e em grande quantidade de domínios, podem se manifestar como sendo *ad hoc*. Adaptações automáticas podem funcionar bem em certos domínios [**RAM 97**] mas podem deixar de funcionar em muitos outros.

Essa abordagem de *resgatar e propor* casos não constitui, porém, a única especialização já feita em relação ao algoritmo geral do RBC. Riesbeck também argumenta em favor de uma outra escola de RBC. Ele considera que, em muitas situações concretas, uma abordagem inversa à do resgate e propositura de casos também pode ser de grande valia. Recomenda então um tratamento de RBC do tipo *seleciona e adapta casos* ("*select and adapt*") [**RIE 96**, p. 384], onde o tempo a ser gasto nestas adaptações faz toda a diferença entre adotar ou não essa abordagem.

2.9 Algoritmos de *ranking* de casos

O processo de busca de similaridade através de métricas tem um subproduto importante. As medidas de similaridade encontradas podem e vêm sendo utilizadas para construir um *ranking* ou ordenamento. Um ordenamento de valores expressa para o usuário o grau de pertinência de uma indagação, ou seja, a pertinência entre uma indagação e os casos resgatados. Esse ordenamento é obtido como se descreve na seqüência.

2.9.1 *Ranking* ou ordenamento

Ranking, por conseguinte, é o processo de RBC que consiste na ordenação daqueles casos que tenham sido parcialmente emparelhados de acordo com as suas pertinências ou prováveis utilidades para o usuário. Este processo de *ranking* de casos é tão característico da tecnologia de RBC que tem, inclusive, servido para distinguí-la de outras tecnologias de agentes inteligentes que não se preocupam em ordenar para o usuário os resultados dos processos de busca de informação. Muitos sistemas apenas anunciam coisas, por exemplo, do tipo: "*Sorry, no match*" ou ainda "*100 matches found*", sem, porém, se preocuparem em propor uma ordem de preferência clara, em relação à indagação feita pelo usuário. No *ranking* do RBC, ao contrário, existe uma ordenação clara de modo a poder guiar o usuário na sua apropriação dos resultados de uma consulta.

O algoritmo que segue usa todos os casos que parcial ou completamente venham a se emparelhar com uma lista dada de pares <*atributo*, *valor*> e faz o *ranking* desses casos de acordo com a quan-

tidade dos emparelhamentos ocorridos. Para cada $caso_i$ guardado na variável *casos-resgatados*, os seus pares <*atributo*, *valor*> são confrontados com os pares <*atributo*, *valor*> do novo problema. As quantidades de emparelhamentos de todos os casos resgatados são então armazenadas em uma variável do tipo *array* chamada *emparelhamentos*. Os casos, portanto, são ordenados com base na quantidade desses emparelhamentos. Os casos que apresentarem a maior quantidade de emparelhamentos serão considerados como sendo aqueles casos mais relevantes para o usuário.

Ranking de casos

begin

Inicializa $emparelhamentos_i$ com 0 em todos os $casos_i$;
Para cada $caso_i$ em *casos-resgatados*
    Para cada par de <$atributo_j$, $valor_j$> no novo problema
        Obter o $valor_i$ do $atributo_j$ no $caso_i$;
        Se $valor_j = valor_i$
        então
            some 1 ao valor de $emparelhamentos_i$ do $caso_i$;
Achar o maior dos $emparelhamentos_k$ entre os ocorridos;
Obter o rótulo do caso em *casos-resgatados* correspondente a $emparelhamentos_k$;

end

2.9.2 Métodos de *ranking*

Dois procedimento, basicamente, têm sido mais usados em *ranking* de casos: *(i) ranking* sem similaridades; *(ii) ranking* com similaridades. O primeiro procedimento funciona em sistemas cujo resgate esteja baseado na quantidade de índices que venham a se emparelhar (número de *matches* entre índices). Ou seja: funciona em sistemas com resgate baseado, por exemplo, no algoritmo seqüencial modificado, da seção 2.7.4. Um *ranking* parcial, nesta hipótese, pode ser obtido resgatando-se e exibindo-se todos os casos – em ordem decrescente – segundo a quantidade de índices que se emparelhem. Assim sendo, todos aqueles casos com n emparelhamentos de índices serão ordenados à frente daqueles casos com $(n-1)$ emparelhamentos de índices, e assim por diante até a inclusão daqueles casos que não possuam quaisquer índices em comum com a indagação do usuário.

No segundo método, porém, o *ranking* depende da medida de similaridade encontrada, sendo que a estratégia consiste em resgatar todos aqueles casos cuja similaridade *SIM* entre indagação-casos exceda um certo limiar k que seja dado. Nesta hipótese, o resultado da aplicação da métrica de similaridade viabiliza o *ranking* que pode ser obtido através da apresentação ao usuário dos casos resgatados - em ordem decrescente da similaridade *SIM*.

2.10 Avaliação e validação de casos

Finalizamos estes fundamentos sobre a tecnologia do RBC apontando ainda o problema da *qualidade* dos casos de uma base. Nem todas as soluções obtidas em consultas a uma base de casos são consideradas pelos usuários como sendo igualmente consultas boas e relevantes; daí a necessidade de se criar mecanismos para estimar o quanto uma base de casos, de fato, esteja a cumprir aqueles objetivos para os quais ela tenha sido projetada. Comunidades de pesquisa em banco de dados têm se preocupado em construir sistemas que sejam *robustos*, *eficientes*, e *flexíveis*. Mas também estas comunidades têm ainda se concentrado no problema da *qualidade* dos produtos de informação, armazenados nestes bancos de dados [**FIR 95**, **BOR 95**]. Enquanto isto, em RBC estas questões de qualidade de bases de casos têm sido grandemente esquecidas, certamente pela complexidade própria da tarefa.

Avaliar a qualidade de casos e de bases de casos é tarefa complexa por envolver *(i) julgamentos de valor*, *(ii) preferências do usuário*, *(iii) grande quantidade de conhecimentos* (sobre os elementos da avaliação, prioridade desses elementos, interinfluências de elementos, influência destes elementos sobre soluções contidas em casos, etc) e até mesmo envolve *(iv) questões de estética*, como assinala Janet Kolodner [**KOL 93**, p. 543]. Ainda segundo J. Kolodner, em domínios de aplicação do RBC onde inexistem importantes teorizações (como no exemplo de CLAVIER), a avaliação/validação de sistemas requer a validação de cada caso, individualmente. Esta é uma tarefa mais apropriada para aqueles usuários que sejam capazes de pesar o mérito dos componentes de uma base de casos ou de cada caso.

2.11 Conclusão

Neste capítulo, apresentou-se uma parte importante do estado da arte da tecnologia de RBC que complementa as discussões iniciadas no Capítulo 1. Foram destacados, nesta apresentação, todos os processos básicos envolvidos no RBC, e também os seus aspectos teóricos, conceituais e de exemplificações necessários para uma visão geral desta tecnologia. A meta do capítulo é servir de pano de fundo para as discussões subsequentes, relativas às contribuições propostas neste trabalho de pesquisa.

Parte 2

Desenvolvimento das metodologias

Que lições de desenvolvimento podemos extrair do estado da arte da computação de casos que acabamos de explorar nos capítulos anteriores? Nesta Parte 2, problemas ainda em aberto no RBC são identificados e propostas de solução para eles são trabalhadas e analisadas. O Capítulo 3 constitui uma visão geral destas soluções. Posicionado nesta parte da tese, o Capítulo 3 representa - na forma de um *overview* - um esforço de integração das propostas metodológicas a serem detalhadas na parte restante da tese. Os Capítulos 4, 5 e 6 detalham esses modelos metodológicos propostos para o RBC, a partir do seu problema maior de similaridade de casos.

Capítulo 3

Visão geral das contribuições propostas e orientação para *Indagação-Resposta*

3.1 Introdução

O presente capítulo trata de um *overview* dos problemas e das soluções propostas em nossa investigação. A meta então será a de oferecer uma visão geral do trabalho, antes mesmo que seja detalhado, nos demais capítulos, cada método por nós investigado, em uma abordagem *top-down* do tipo "primeiro a floresta, depois as árvores". Nos capítulos 1 e 2 anteriores, foi mostrada uma radiografia do estado atual do raciocínio baseado em casos (RBC) como tecnologia computacional e de modo a incluir tanto os seus principais processos operacionais quanto as relações desta tecnologia com a IA. O presente capítulo parte das sínteses sobre os trabalhos realizados no âmbito da presente investigação, quais sejam:

- Identificação daqueles problemas de RBC ainda em aberto e que foram selecionados como objeto de tratamentos ao longo desta tese;
- Enunciação das soluções encontradas para estes problemas a serem detalhadas oportunamente; e
- Posicionamento destas soluções formuladas em nossa investigação em um contexto mais amplo, qual seja o contexto dos *Sistemas de Indagação-Resposta* (ou *Query-Answering Systems*).

A meta do capítulo, portanto, consiste em colocar juntos problemas de RBC e soluções formuladas em torno de uma visão unificadora destas mesmas soluções.

3.2 Problemas e soluções investigados

Cinco problemas fundamentais do raciocínio baseado em casos foram aqui definidos como sendo, particularmente, de interesse no contexto da presente investigação (excluindo-se o problema de modelagem do domínio selecionado para experimentações). Introduzimos, neste ponto, esses problemas centrais investigados e as suas respectivas soluções:

Problema 1: *Como computar a similaridade entre casos de modo a incorporar significantes componentes qualitativos*?

Este problema, como será visto na seção 4.2.2, constitui reconhecidamente uma tarefa para a qual se torna relevante examinar o modo como as pessoas fazem julgamentos de similaridade entre objetos, como asseverara J. Kolodner (e tudo que se refere a *cognitivo* nesta tese tem justamente este sentido). Aqui, o problema a resolver está em como encontrar, na memória, casos computacionais que mais naturalmente (e não a qualquer custo) se emparelhem uns aos outros.

Problema 2: *Como indexar/representar casos de modo apropriado à metodologia de computação de similaridades proposta acima*?

Este, portanto, é um problema decorrente do anterior. Admitimos, por conseguinte, que a metodologia de computação da similaridade influencia o modo como os casos devam ser estruturados; influencia a metodologia de *indexação* e não vice-versa. Neste particular, os métodos de indexação (apontados na literatura de RBC) são de pouca ajuda porque ora eles são confundidos com métodos de busca, propriamente **[WIL 97]**, ora são reduzidos à mera aplicação de diretrizes (*guidelines*) (cf. seções 2.5.3-2.5.4).

Problema 3: *Como estender a metodologia de similaridade cognitiva para também obter-se o ordenamento ou o ranking flexível de casos?*

O modo convencional como o RBC realiza o *ranking* de casos padece daquilo que estamos a denominar de *tirania do ordenamento final*. Uma vez o algoritmo de *ranking* haver produzido o seu ordenamento dos casos resgatados, não haverá mais opção para o usuário dessa ordenação. A única possibilidade de alterar-se um ordenamento estanque já feito será através da chegada de um novo caso resgatado ou, como em alguns sistemas **[BRO 95]**, através da apresentação de um atributo novo para algum dos casos originais. O nosso problema aqui consiste em estender o mesmo enfoque da similaridade cognitiva para introduzir flexibilidade no *ranking* de casos, permitindo ao usuário ordenar casos conforme os "pontos de vista" desse usuário sobre as similaridades e seus parâmetros.

Problema 4: *Como avaliar uma base de casos, a partir da* QUALIDADE *dos resultados das consultas feitas por usuários*?

Antever-se que a participação do usuário de bases de casos seja vital neste processo de avaliação de sistemas baseados em casos – um problema que também afeta áreas da computação como Banco de Dados e *Information Retrieval*, por exemplo. Enquanto, porém, esta avaliação de performance já está sendo resolvida tanto em Banco de Dados **[FIR 95, BRO 95]** como em *Information Retrieval* **[KOW 97]**, essa mesma questão

continua sendo um problema ainda pouco visitado na comunidade de raciocínio baseado em casos, tanto teórica quanto operacionalmente **[KOL 93,** p. 540].

Problema 5: *Como integrar as soluções propostas para os problemas 1, 2, 3 e 4, incorporando-as em um contexto mais amplo de preocupações do RBC?*

Em outros termos: como unificar as metodologias propostas orientando-as para um problema mais amplo do RBC? Essas metodologias foram elaboradas para funcionarem em diferentes contextos de uso do RBC. A meta, porém, ao definir-se o Problema 5, é demonstrar que o conjunto das soluções particulares aqui propostas está apropriado e pode ser orientado para também resolver um outro problema – de muito maior abrangência – qual seja aquele problema proposto por C. K. Riesbeck **[RIE 96]** e que consiste em se enxergar base de casos como *base de respostas* a indagações (Cap. 1, seções 1.4.3.2 e 1.4.3.3).

Correspondendo a cada problema acima, a nossa investigação buscou formalizar o seguinte conjunto de soluções metodológicas que, no presente capítulo, são orientadas para o problema da Indagação-Resposta por computador:

Solução 1: *SIM(m,p) – Similaridade de casos baseada na teoria de Tversky-Gati*

Por esta metodologia de similaridade, um caso em RBC precisa ser caracterizado através da representação dos seguinte componentes: *(i)* atributos; *(ii)* diagnosticidade de atributo; *(iii)* valor de atributo; *(iv)* atributos compartilhados por casos; *(v)* atributos não compartilhados; *(vi)* importância de atributos compartilhados; *(vii)* importância de atributos não compartilhados. Diferentemente de outras abordagens mais simples, a metodologia para a computação da similaridade $SIM(m,p)$ é elaborada de modo a levar em conta todos estes componentes qualitativos de casos e se fundamenta na explicitação, por nós, da função de Tversky dada por $sim(i,p) = \theta f(I \cap P) - \alpha f(P - I) - \beta f(I - P)$, onde $\theta , \alpha , \beta \geq 0$ **[MAR 97, MAR 99b]**.

Solução 2: *IBT – Indexação de casos baseada em tabela*

Para acomodar todos estes componentes no interior dos casos, propõe-se um modelo de organização de índices denominado *Indexação Baseada em Tabela* (*IBT*). O método, em parte, é discutido em *Teoria de Agentes* **[RUS 95,** p. 300], porém sem qualquer relação direta com a tecnologia de RBC. Através dele, cada característica de um caso deve ser representada por termos com uma entrada nesta tabela. Por outro lado, os respectivos valores para os termos descritores de atributos eliciados podem ser obtidos quer através de especialistas, quer através de métodos quantitativos tais como o *credit scoring* ou quaisquer outros **[MAR 99c]**.

Solução 3: *ORDEN – Ordenamento ou ranking flexível de casos*

A idéia básica nesta metodologia *ORDEN* é usar a própria métrica da similaridade (o seu *pré-processamento* na forma $aX - bY - cZ$) para gerar a discriminação dos casos a serem ordenados. Haverá tantos ordenamentos de um mesmo conjunto de casos quantos forem os pontos de vista do usuário sobre a similaridade buscada e sobre os seus parâmetros ($a, b, c \geq 0$). Se a ênfase dada pelo usuário recair sobre aqueles atributos comuns a dois casos em comparação, o usuário fará então variar o valor do parâmetro a durante o pré-processamento da similaridade. Cada valor de a na fórmula do pré-processamento vai dar origem a uma nova medida de similaridade e, conseqüentemente, dar origem a um novo reordenamento dos casos. Porém, se os pontos de vista do usuário privilegiarem os atributos não compartilhados, ele então fará variar os parâmetros b, c também dando origem a diferentes similaridades e ordenamentos sobre um mesmo conjunto de casos **[MAR 99b]**.

Solução 4: *AVAL – Avaliação qualitativa de casos baseada em métricas*

Sistemas baseados em casos acessam um espaço de casos e retornam um conjunto deles em resposta a uma indagação do usuário. Nem todas estas respostas, porém, costumam ser igualmente aceitas como úteis pelo usuário. Para acessar a *qualidade* das respostas contidas em uma base de casos, a solução proposta em *AVAL* pressupõe três aspectos sobre o funcionamento destas bases, que são:

- admitir que a *qualidade total* da base vai depender da qualidade de cada caso potencialmente resgatável, independentemente de seus mecanismos de manipulação;
- admitir a contabilização – manual ou automaticamente – da qualidade das respostas a partir das consultas feitas à base;
- usar esta contabilização para construir métricas de avaliação, com prioridade para a *métrica de precisão* e para a *métrica de retorno* ou *relembrança.*

Solução 5: *I-RBC – Indagação-Resposta Baseada em Casos*

Em relação à solução do problema 5, mostramos como o paradigma de *Query-Answering Systems* (representado sobretudo pela *Escola Escandinava*) pode se constituir um paradigma aglutinador ou um guarda-chuva para as soluções 1, 2, 3, e 4 de tal modo a poder grupar, sob este guarda-chuva, as metodologias de RBC aqui propostas. Uma tal visão concebida para integrar as nossas contribuições se fundamenta sobre os seguintes pressupostos **[MAR 99a]**:

- A *explicação automática* – em um de seus paradigmas – pode ser obtida sem necessariamente se ter de recorrer a cadeias lógicas de inferências. São explicações veiculadoras de *elucidação* que são obtidas não por inferências (Explicação do tipo 1), mas por *combinação de ingredientes* (Explicação do tipo 2).
- *Respostas* à indagação são definidas – em nossa concepção integradora – como sendo explicações do tipo 2, como será visto na seção 3.5.1.2.
- Explicações podem ser representadas por *casos* (segundo J. Kolodner). Supondo-se que estas explicações representáveis por casos sejam explicações do tipo 2, conclui-se que as *Respostas* de interesse para nossa abordagem também podem ser representadas por casos do tipo que estamos a modelar. Ou seja, representadas por casos cuja capacidade de *oferecer respostas* vai decorrer da capacidade que possuem de elucidar situações.
- As soluções 1, 2, 3, e 4 propostas para casos, podem finalmente ser colocadas a serviço desta visão de *Indagação-Resposta Baseada em Casos* (*I-RBC*) – que estamos a propor como sendo um elo entre o RBC e os *Query-Answering Systems* ora, paralelamente, em desenvolvimento.

Esta *Solução 5* que está na base da presente *overview* passa então, daqui pra frente, a ser o objeto de estudo da parte restante deste capítulo. Ou seja: passamos a examinar no capítulo o problema de como estabelecer um relacionamento entre RBC e a área de sistemas *Query Answering* de modo a viabilizar cenários de Indagação-Resposta capazes de fazerem uso das demais contribuições a serem detalhadas oportunamente.

3.3 RBC e *Query-Answering Systems*

3.3.1 Motivações e razões

Sistemas de *Indagação-Resposta* ou I-R (do inglês, *Query-Answering Systems*) estão bem caracterizados no conceito amplamente discutido por Troles Andersen **[AND 97]**:

> *Sistemas inteligentes de indagação-resposta são sistemas computacionais capazes de transformar a indagação de um usuário – expressando para uma base de conhecimentos as suas necessidades – em uma resposta que venha a conter informação útil.*

São estes os sistemas que estamos a procurar interligar com o RBC, utilizando-se para isto o conjunto das proposições por nós formuladas. Três razões básicas fundamentam a busca por uma conexão possível entre o RBC e a área de Indagação-Resposta compreendida tal como na definição

acima:

- *Carência de abordagens de RBC na comunidade de Query-Answering.* A produção científica da Escola Escandinava de sistemas *Query-Answering* parece representar o que existe de mais significativo nesta área. Nesta produção, no entanto, não nos foi possível identificar abordagens que façam uso da tecnologia de RBC para fins de processamento de Indagação-Resposta, como se pode ver pelo teor dos trabalhos apresentados nos workshops FQAS'94, FQAS'96 e pelos trabalhos coordenados por Andreasen **[AND 97]**. Daí, talvez, o interesse demonstrado quando da nossa proposição de uma abordagem deste gênero por ocasião do 6th International Workshop on Knowledge Representation Meets Databases (Linköping, Sweden, 1999) **[MAR 99a; FRA 99]**.

- *Carência de abordagens de Query-Answering na comunidade de RBC.* Também tudo está a indicar a carência desta abordagem dentro da comunidade de RBC. A tecnologia de RBC pode ser facilmente identificada como sendo de: *(i)* planejamento; *(ii)* projeto; *(iii)* interpretação; *(iv)* classificação; e *(v)* resolução de problemas. Porém, não será tarefa fácil identificar o seu emprego para sugerir respostas a indagações. A este respeito, existe, como visto na seção 1.4.3.3, a indicação por parte de Riesbeck não só de que se trata de uma tarefa relevante mas também uma tarefa urgente. A tendência de desenvolvimento futuro do RBC – segundo Riesbeck – passará pelo seu emprego na tarefa direta de responder questões do usuário **[RIE 96]**.

- *Interesse industrial.* Uma outra razão a motivar a busca da conexão RBC e *Query Answering* é o interesse industrial pela questão. Este interesse foi recentemente expresso por Bill Gates ao escrever sobre "*A Empresa na Velocidade do Pensamento*" e que analisamos na seqüência **[GAT 99**, p. 235**]**.

3.3.2 Aceleração de respostas: Bill Gates (1999)

Metodologias para responder indagações constituem uma das preocupações de Gates **[GAT 99]** e, por extensão, uma preocupação de caracter industrial, sobretudo ao se levar em conta as limitações atuais destes sistemas. O autor se preocupa sobremodo com os métodos denominados por ele de métodos de *aceleração de respostas*. Em seu levantamento, o autor não chega a mencionar qualquer papel para o RBC a respeito da aceleração de resposta. O autor, porém, destaca e analisa todo o esforço que vem sendo feito no sentido de desenvolver ferramentas digitais para levar, por exemplo, o paradigma de banco de dados a superar as suas limitações relativas ao modo de oferecer respostas a indagações do usuário. Entre outras ferramentas, foram destacadas:

- *Tabelas dinâmicas* ("pivot tables"). Estas tabelas (parecidas com planilhas eletrônicas)

possibilitam que um mesmo conjunto de dados possa ser visto sob múltiplos e variados ângulos.

- *Modelos* ("templates"). São ferramentas que permitem a montagem de dados em apreço em formatos padronizados, sendo possível gerar relatórios flexíveis e personalizados para atender às necessidades específicas dos usuários.

- *Depósitos de dados* ("data warehouse"). Cada banco de dados individual de uma empresa usualmente costuma ter capacidades limitadas em relação ao tempo de resposta (podendo levar de trinta a quarenta minutos para sua obtenção, conforme Gates) e também capacidades limitadas na geração de relatórios (geralmente restritas ao pessoal técnico das empresas, conhecedor de linguagens de banco de dados). *Data warehouse* permite combinações com tabelas dinâmicas para ampliar o acesso a dados e permitir assim indagações com posterior refinamento/detalhamento.

- *Garimpagem de dados* ("data mining"). São metodologias embasadas em algoritmos que permitem *descobrir padrões* úteis em grandes quantidades de dados e que tornam as indagações muito mais eficientes. Dados originalmente coletados para a contabilidade ou para a escrituração contábil, por exemplo, passam a ser reconhecidos (pelos algoritmos de garimpagem) como mina potencial de informações para finalidades outras como a modelagem, as previsões e o apoio à decisão.

- *Aceleração de resposta via Web* ("PRINCESS"). Gates se preocupa sobremodo com a necessidade da *aceleração de respostas* para os atendimentos a clientes em contextos de negócios. Ele examina detalhadamente o papel da ferramenta japonesa PRINCESS, baseada na Web. Trata-se de ferramenta para suporte a indagações feitas por médicos e farmacêuticos, no domínio de produtos farmacêuticos (ou seja: para respostas em ambiente de *help desk*). Utilizando mecanismos de pesquisa em tempo real e armazenamento óptico, a ferramenta permite descobrir respostas para aquelas perguntas mais difíceis e responder imediatamente a cerca de metade de todas as indagações que chegam dos clientes usuários do *help desk*. Respostas não oferecidas prontamente são atendidas dentro de um certo prazo combinado.

3.4 Base conceitual sobre *Query-Answering*

Sistemas modelados para a tarefa específica de responder indagações envolvem numerosos problemas computacionais tais como:

(i) Qual deve ser o *conteúdo* de uma indagação e o seu possível modo de formulação?

(ii) Como tornar flexível para o usuário o emprego dos sistemas de I-R?

(iii) Como implementar os mecanismos de obtenção de respostas e também os necessários mecanismos de comparação de itens de informação, ou mecanismos de similaridade?

(iv) Como avaliar a *utilidade* das respostas geradas pelo sistema?

(v) Como acelerar a velocidade das respostas que se espera de um sistema?

(vi) Como estimar a *qualidade* das respostas já oferecidas por um sistema?

(vii) Que linguagens nebulosas de indagação (*fuzzy query languages*) podem ser empregadas por um usuário?

(viii) Como formular a um sistema computacional indagações com requerimentos ou *indicação de tempo*?

Este tem constituido o programa ou o estatuto da pesquisa computacional sobre *Indagação-Resposta*, no contexto da Escola Escandinava a que nos referimos anteriormente. Examinamos na seqüência conceitos relativos a essa problemática antes de posicionarmos, neste contexto, as metodologias de RBC desenvolvidas na presente investigação.

3.4.1 Paradigmas de sistemas *Query-Answering*

Considere-se que se queira construir um sistema para aceitar e responder indagações tal como um agente de viagem. Dentro do leque das possíveis indagações que um usuário possa querer ver respondidas, estão aquelas indagações que abaixo exemplificamos. São indagações possíveis em sistemas agentes de viagem e que comumente envolvem três níveis típicos de complexidade. Três paradigmas computacionais de obtenção de respostas aceitariam diferentes indagações e também diferentemente as tratariam **[POR 89; AND 97]**:

- *Bancos de dados convencionais* [Qual o horário de partida do vôo VAR477?]

  Este é o tipo de resgate de informação normalmente associado a *bancos de dados* convencionais. A resposta que se fizer necessária vai ser constituída por uma porção de informação, diretamente vinculada àquela informação contida na indagação.

- *Bancos de dados dedutivos* [Posso viajar aos sábados de Teresina a Campos do Jordão?]

  Já esta classe de indagações requer algo mais do que uma mera e pré-definida entrada em um banco de dados. São indagações a exigirem que todas as possibilidades de conexões de viagens e suas respectivas condições sejam visitadas. Responder indagações como estas requer processos de dedução e constitui o tipo de indagação que podemos associar aos *bancos de dados dedutivos* **[KUM 92]**.

- *Sistemas intensivos de conhecimento* [Qual o melhor roteiro que posso fazer com $2.000?]

  Por fim, esta classe de indagação pertence a um tipo muito mais complexo de tratamento da indagação-resposta. Aqui, a abrangência das possibilidades é tão vasta que a indagação vem a requerer complexa interação com o usuário e inferências de múltiplos níveis que são características próprias dos *sistemas intensivos de conhecimentos*. Tudo está a indicar que estamos ainda distantes da completa exploração industrial de tais sistemas inteligentes de indagação-resposta.

3.4.2 Propriedades de respostas

Em teoria, importantes propriedades nestas classes de tratamento de indagações são [**POR 89**]:

- *Resposta à indagações situadas*. Indagações costumam representar demandas bem situadas em relação ao usuário. No caso do agente de viagem, por exemplo, esta característica "situacional" da indagação poderá estar presente na forma de exigências tais como: Qual o melhor roteiro (em relação ao sujeito da indagação)? Onde começa o roteiro (em relação ao espaço)? Quando no ano (em relação ao tempo)?

- *Resposta parcial*. O sistema pode não dispor de todas aquelas informações requeridas para uma completa e boa resposta ao usuário. Mesmo assim, respostas parciais resultantes de emparelhamentos parciais (*partial matching*) por parte de um sistema podem ser de grande valia para esse usuário.

- *Resposta interativa*. Dadas as características de paradigmas e de domínios, sistemas de indagação-resposta podem requerer do usuário engajamentos em processos interativos. Neste caso, uma resposta interessante ao usuário poderá ser não apenas aquela derradeira informação advinda do sistema, mas todo aquele conjunto de resultados intermediários originados durante um processo de interação.

- *Resposta útil*. O que será uma resposta útil para o usuário? Pode-se antever que, em qualquer indagação formulada a um sistema, existirá geralmente a possibilidade de muitas e diferentes respostas. Encontrar não apenas uma resposta qualquer entre várias outras respostas possíveis mas aquela que melhor venha a satisfazer o indagador, constitui o problema central destes sistemas.

Segue um exemplo dessa utilidade da resposta computacional esperada por um usuário.

3.4.3 Utilidade de respostas

Considere-se o seguinte exemplo de indagação do ponto de vista da utilidade das respostas a que dá origem **[POR 89]**:

Que computadores estão sendo usados em pesquisas de IA?

Respostas possíveis para esta indagação vão se enquadrar dentro daqueles paradigmas da seção 3.5.1. Respostas extremas seriam:

(i) VAX 2000 Mod.87E460565, VAX 2000 Mod.87B460565, ...

(ii) Os computadores que estão sendo usados em pesquisas de IA

A utilidade da resposta *(i)* é típica de bancos de dados convencionais. Consiste tão somente na enumeração de todos os elementos individuais que satisfaçam aquelas condições/restrições presentes na indagação. A resposta *(ii)*, por outro lado, é extrema mas pode ocorrer no contexto do paradigma lógico. Este tipo de resposta não oferece qualquer *ganho de informação* em relação à indagação feita, embora possa estar logicamente correta ou ter sido derivada por algum processo lógico de inferência. Respostas úteis para a mesma indagação inicial poderiam ser:

(iii) "Máquinas convencionais, mas com grande memória";

(iv) "Máquinas paralelas"; ou ainda,

(v) "Todos os computadores, exceto, microcomputadores".

Respostas úteis costumam ser compactas – por conterem descrições mínimas – mas também costumam ser claramente portadoras de *ganho de informação*, como no exemplo apresentado.

3.5 Posicionamento das contribuições em cenários de *Indagação-Resposta* baseados em casos (I-RBC)

Tendo acima caracterizado os ambientes de Indagação-Resposta com as propriedades que acabamos de examinar, daqui pra frente neste capítulo, mostramos como eles podem ser concebidos com a ajuda das nossas contribuições. As nossas soluções de RBC foram apenas introduzidas na seção 3.3 (*IBT*, *SIM(m,p)*, *ORDEN* e *Aval*) – justamente com esse objetivo de poderem ser referenciadas e acopladas à concepção de um modelo de I-R que estamos a denominar de modelo de *Indagação-Resposta Baseado em Casos* (I-RBC) **[MAR 99a]**.

Uma visão geral integradora tanto das nossas metodologias particulares quanto dos aspectos de I-R já abordados está representada na Figura 3.1.

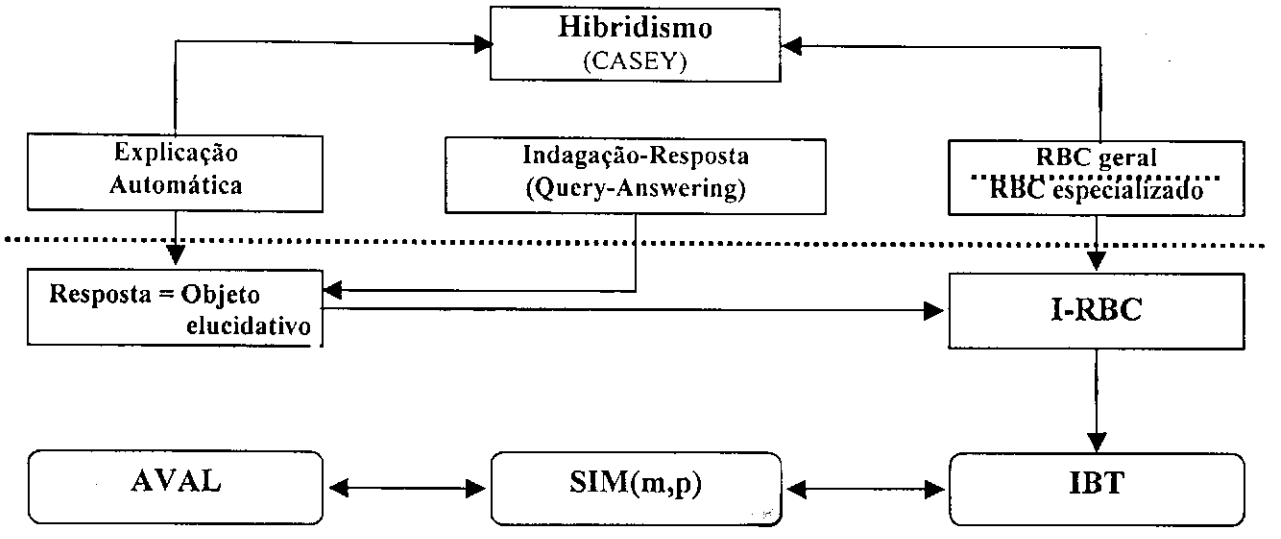

Figura 3.1: *Overview* das contribuições ao RBC e orientação para *Indagação-Resposta*

A Figura 3.1 mostra, portanto, uma *overview* de todo esse trabalho. Ela ilustra tanto a origem da concepção *I-RBC* quanto os seus componentes funcionais necessários a sua realização computacional. Duas partes distintas estão separadas pela linha tracejada, na figura.

A parte superior indica os três domínios da IA cujas imbricações estão a contribuir para a formação deste *framework*, apesar destes domínios parecerem se desenvolver completamente em paralelo dentro da IA: *(i)* explicação automática; *(ii) query-answering* (introduzido acima); e *(iii)* RBC (inclusive as suas especializações). A única interligação historicamente já estabelecida entre estas áreas da IA e que nos foi possível detectar na literatura é aquela representada pelo *hibridismo* do sistema CASEY (no retângulo superior da Figura 3.1). O sistema CASEY dedica-se ao problema da criação de explicações e seu hibridismo reside, justamente, na associação de apenas dois dos paradigmas constantes na figura: explicação automática (via regras) e via RBC, porém, sem qualquer influência do paradigma de *Query –Answering*.

A parte inferior da figura, por outro lado, mostra o interesse em nossa investigação em estabelecer conexões entre estes diferentes paradigmas paralelos de IA, sendo a concepção I-RBC o resultado destas conexões encontradas. O *rationale* subjacente ao modelo I-RBC, por conseguinte, é resultante da descoberta que fizemos de três *imbricações* conceituais fundamentais e que vamos denominar de:

- Conexão *Explicação - Resposta* (seção 3.6.1);
- Conexão *Resposta - Caso* (seção 3.6.2); e a
- Conexão *Aceleração de Resposta - RBC Especializado* (seção 3.6.3)

Detalhamos na seqüência estas conexões conceituais formadoras da concepção I-RBC em apreço.

3.5.1 Conexão explicação – resposta

O primeiro fundamento sobre o qual se apoia a nossa visão de I-RBC foi estabelecido a partir da *imbricação* explicação-resposta: toda *explicação* constitui uma *resposta* mas nem toda *resposta* constitui uma *explicação*. Respostas ambíguas, não elucidativas e que estejam fora de contexto, não constituem explicações, por exemplo.

Existe, porém, uma classe de *respostas* que coincide com *explicações* – não apenas em relação a seus conteúdos informativos mas também em relação ao método de obtenção. As explicações podem ser classificadas de diferentes modos. Interessa-nos, neste ponto, a sua classificação em dois grandes grupos quanto aos métodos de obtenção [**TAN 91,** p. 51]:

- Explicação de tipo 1: *por introspecção*;
- Explicação de tipo 2: *por moldagem*

3.5.1.1 Explicação de tipo 1

Na *explicação por introspecção*, o programa examina o seu próprio conhecimento embutido e a sua memória de soluções de problemas para explicar-se a si próprio. A partir do exame de suas próprias regras em sua base de conhecimento (a introspecção), o sistema *deriva* explicações. O exemplo clássico desta classe de explicação é aquela embutida nos sistemas expertos baseados em regras. A explicação no sistema CASEY, não obstante empregar o RBC, também pertence a esta classe [**KOL 93**]. O sistema tem de inspecionar suas *regras de evidência* e *regras de adaptação* para poder *derivar* explicações. Portanto, a introspecção leva sempre a algum processo de derivação capaz de *inferir* explicações. Diferentemente desta classe de explicação mais antiga, a explicação que vai biunivocamente equivaler-se ao conceito de resposta – e por isso mesmo vai interessar à nossa visão de indagação-resposta – é a explicação de tipo 2.

3.5.1.2 Explicação de tipo 2

As explicações, em muitas situações, são concebidas, diferentemente das explicações por processos de inferências, como estruturas moldadas ou modeladas ("*concocted*", na expressão inglesa de Tanner e Keuneke [**TAN 91** p. 51]). Isto é, explicações feitas por combinação de diferentes componentes ou ingredientes:

> [*Explicações por Moldagem*] *não tentam estabelecer relações (*INTROSPECÇÃO*) sobre como as decisões são alcançadas em um sistema, mas elas por si próprí-*

> *as tornam* PLAUSÍVEIS *essas decisões ao serem exibidas ao usuário. Esta classe de explicação ou* JUSTIFICAÇÃO *se torna necessária sempre que os sistemas não tenham acesso aos seus próprios registros de solução de problemas ou ainda quando a informação contida nesses seus registros não esteja transparente para o usuário ou, de alguma forma, esteja incompreensível para este usuário".*[2]

Provas matemáticas, por exemplo, esclarecem e até persuadem sem que seja necessário se reportar a todo um processo sobre como os matemáticos *derivam* teoremas **[TAN 91]**. A explicação de tipo 2, portanto, em vez de ser conseguida via inferências/derivação – como se tornou conhecida a explicação automática – é antes obtida por *combinação de componentes*.

O conceito de resposta tomado na concepção I-RBC tem esse sentido de explicação de tipo 2, quanto à sua obtenção. Por outro lado, a área de *Query-Answering* vê a resposta computacional fundamentalmente como um objeto capaz de fornecer elucidações **[AND 97]**. Com base nestes conceitos, introduzimos então o nosso primeiro alicerce para a abordagem aqui formulada:

> *Resposta é um objeto capaz de veicular ganhos de informação para o usuário e que seja um objeto construído por combinação de elementos mais simples, em vez de obtido por derivação ou inferências.*

Tal conceito, portanto, corresponde ao nexo resposta-explicação por nós detectado.

3.5.2 Conexão resposta – caso

Uma segunda base conceitual para a concepção I-RBC é a descoberta de que *respostas*, por sua vez, podem ser pensadas em termos de *casos* do RBC. A conexão entre *respostas* e *casos* pode ser estabelecida ao se examinar uma imbricação anterior já feita por Janet Kolodner entre explicação e casos **[KOL 91**, p. 6]:

> *"Uma característica que freqüentemente se espera dos sistemas de resolução de problemas é a sua habilidade de oferecer explicação para alguma solução obtida [a obter]. Em sistemas de RBC, tais* explicações são SIMPLESMENTE O CASO (OU CASOS) *que for/forem usados, tornando estas explicações fáceis de serem "geradas". De fato – dado que o RBC resolve problemas do mesmo modo como as pessoas resolvem – uma explicação baseada num caso concreto passado pode ser mais satisfatória do que explicações originadas de cadeias de regras, que é o método mais antigo de explicação nos sistemas baseados em regras".*

Kolodner, ao pronunciar a expressão "*gerar explicação*", não quer, de modo algum, fazer referência a quaisquer métodos de "explicação por inferências" e a encadeamentos de regras, de qualquer natureza, mesmo as de adaptação. Seu "gerar explicação", ao contrário, significa poder resgatar e exibir casos que falem por si (casos explicativos). É este sentido kolodneriano de "gerar" como

[2] Sublinhamentos nossos.

resgate & exibição de coisas que falem por si só e que estamos tomando de empréstimo para associarmos ao conceito de resposta no modelo I-RBC.

Ora, se explicações podem ser representadas como casos do RBC e, por sua vez, respostas são construídas como explicações de tipo 2, então respostas podem ser apropriadamente representadas como casos do RBC. Esta é a ilação fundamental subjacente à concepção I-RBC representada na figura 3.1. O raciocínio desenvolvido até este ponto pode ser seguido através da Figura 3.2.

A moldagem de casos desenhados para funcionarem como respostas (*casos-resposta*) se dará, por conseguinte, por *composição de elementos* elucidativos. Como então obter-se esta composição de elementos? Processos de indexação de casos, em geral, e em particular a indexação do tipo *Indexação Baseada em Tabela* (***IBT***, no modelo da Figura 3.1) vão garantir este processo de obtenção de elementos constituídos por atributos, seus valores, pesos e *diagnosticidades* para moldar respostas em um domínio particular (de acordo com a proposta de indexação a ser detalhada no Capítulo 5).

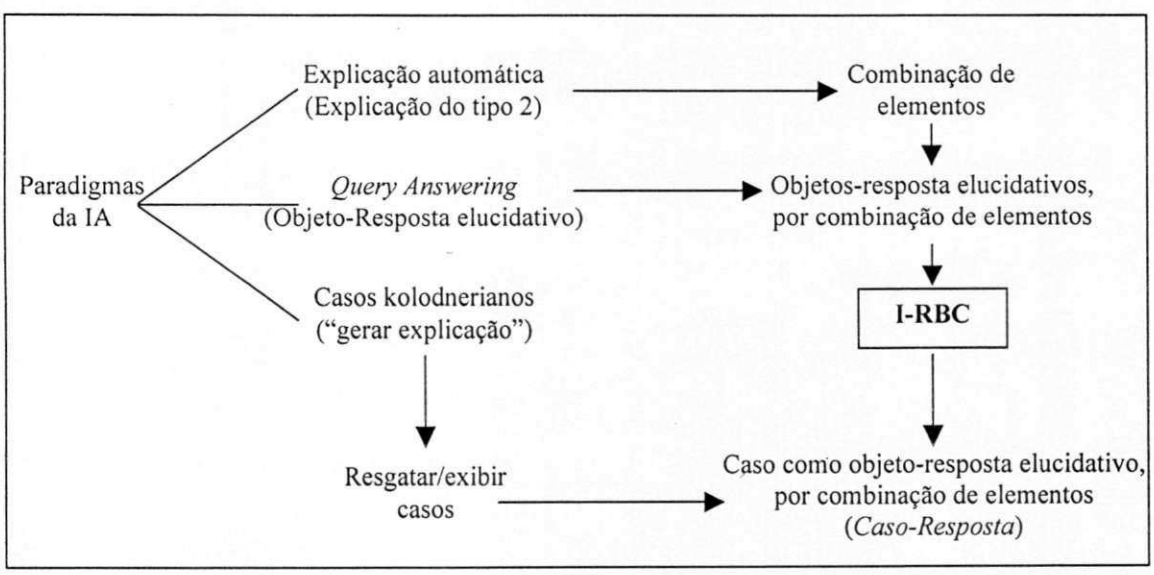

Figura 3.2: Formação do conceito de *Resposta* como objeto elucidativo em I-RBC

3.5.2.1 Resgate de respostas: exemplo dos garçons

Casos-resposta, portanto, são objetos elucidativos relacionados à uma certa *indagação* (*Query Case* e *Response Cases* [**MAR 99a**]). Ao fazer indagações, qual então a expectativa de um usuário que espera resgatar respostas na forma de casos? A este respeito, garçons em restaurantes podem oferecer uma boa ilustração sobre as propriedades dos objetos que estamos a definir como sendo casos-resposta:

> *O primeiro dos garçons traz para a mesa do seu cliente exatamente aquela marca de vinho tal qual foi pedida pelo cliente, mesmo sabendo ele haver o seu*

> *cliente anteriormente já ordenado uma refeição incompatível com aquele vinho. Um segundo garçom, porém, na mesma situação, se comporta diferentemente. Em vez de cegamente trazer só aquele vinho inapropriado para a refeição pedida, ele traz duas taças juntas ao vinho encomendado ao tempo em que "lembra" ao cliente sobre a existência de um vinho melhor para acompanhar sua refeição.*

Assim também devem funcionar os sistemas de indagação-resposta baseados em casos, como examinamos na seqüência.

3.5.2.2 Resgate de respostas: propriedades

O exemplo dos garçons ilustra a propriedade de que uma resposta não necessariamente tem de corresponder – de um modo exato – a uma interpretação daquelas restrições contidas na indagação, na forma do comportamento do primeiro garçom. Ao contrário, o sistema ideal deve ser capaz de exibir respostas úteis mesmo quando para atender indagações que tenham sido sub-especificadas, por exemplo. Ou seja, resgates qualitativamente úteis ao interrogador são aqueles que ocorrem sobretudo quando:

- os objetos, em algum sentido, forem similares àqueles descritos em uma indagação; ou quando
- os objetos só parcialmente se conformarem às restrições veiculadas em uma indagação; ou mesmo quando
- os objetos simplesmente costumarem vir acompanhados com aqueles outros objetos diretamente associados a uma indagação.

A função de similaridade ***SIM(m,p)*** na Figura 3.1 tem este papel de descobrir a similaridade dos objetos-resposta. Enquanto isto, a proposta ***AVAL*** (também indicada na Figura 3.1) tem o papel de avaliar a qualidade dos objetos-resposta oferecidos ao usuário, em relação ao resgate ideal aqui discutido.

3.5.3 Conexão aceleração de resposta – RBC especializado

Uma terceira e última conexão estabelecida para fins desta formulação de I-RBC diz respeito a uma cláusula de *aceleração de respostas* – conforme a idéia trabalhada por Gates (e descrita na seção 3.4.2). A conexão, por nós descoberta, é a de que a especialização do RBC do tipo "*resgata e propõe*" (*Retrieve and Propose*) (cf. seção 2.8.3) vem justamente em apoio a esta necessidade de aceleração de respostas. O RBC especializado, neste sentido, vai contrapor-se a metodologias mais tradicionais em IA do tipo *gera e testa* (*Generate and Test*). Esta condição de que a nossa aborda-

gem deva adotar a aceleração de resposta está indicada pela seta que parte do *RBC especializado* para o retângulo representando a proposta *I-RBC*, em análise. Examinamos então, na seqüência, as diferenças entre esta visão tradicional do gerar e testar em IA e a visão decorrente da conexão entre aceleração de resposta e RBC especializado que estamos a estabelecer.

3.5.3.1 "*Gerar e testar*"

"*Gerar e testar*" refere-se a um método fundamental de busca que requer dois programas auxiliares:

- um *gerador* de soluções candidatas e
- um *testador* de soluções,

onde a *saída* do gerador serve de *entrada* para o testador. Se um teste tem sucesso, o testador saí então com alguma solução. Caso contrário, o programa gerador deverá produzir uma outra solução candidata. Se o programa gerador não conseguir produzir esta outra solução candidata, ele próprio terminará seu processamento pela via do insucesso. Uma representação esquemática deste método está descrita na Figura 3.3.

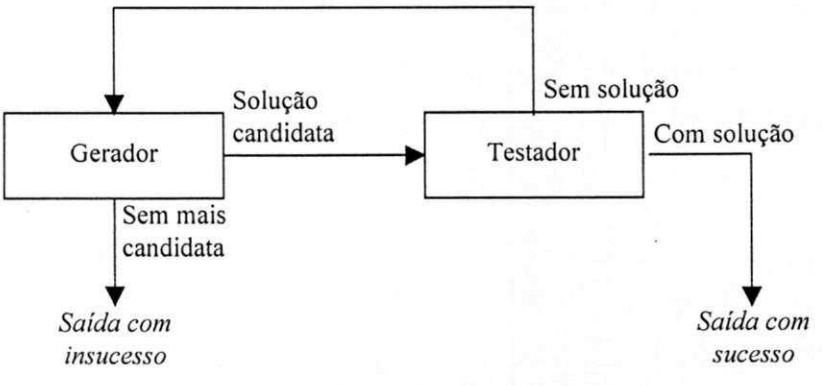

Figura 3.3: Método para *gerar e testar* soluções

Para realizar-se uma busca do tipo *gera e testa*, um dos algoritmos mais simples será este algoritmo abaixo.

Busca do tipo gerar-e-testar

begin

Enquanto o gerador não estiver vazio faça:
    Se testador-solução(gerador(candidata)),
    então
        o processo termina com sucesso notificando qual seja a solução candidata;
    senão /* Aqui quando nenhuma solução for encontrada */
        o processo termina com insucesso;

end

Sistemas projetados sob o enfoque de gerar e testar soluções, porém, podem apresentar desvantagens. Os sistemas, por exemplo, têm de prever um conjunto finito de desvios alternativos que eles devem seguir – por exemplo, os desvios "*x*", "*y*", "*z*" a dependerem das soluções geradas **[RIE 96]**. Nesta hipótese, tais sistemas terão de gerar ou construir estruturas representacionais a serem examinadas naquele momento do emprego da solução. Isto significa que estes sistemas têm também de providenciar o mapeamento destas estruturas que representam soluções com os desvios "*x*", "*y*", "*z*". Necessariamente, tais mapeamentos acarretam a exigência de tempo extra tanto para a execução do sistema quanto para a sua manutenção. Uma segunda dificuldade, além deste mapeamento referido, diz respeito ao próprio controle deste tempo. É muito difícil controlar ou impor limites de tempo em processos de geração (quer a *geração por refinamentos*, quer a *geração por ajuntamento*). Se vier a ocorrer uma parada prematura deste processo de geração obter-se-á ou uma solução incompleta para o problema ou uma solução não funcional.

3.5.3.2 "*Selecionar ou descobrir*"

De modo geral – a depender do *design* do sistema e de sua aplicação – pode ser muito mais útil para um sistema *selecionar ou descobrir* soluções já indexadas e armazenadas do que gerar e testar. Gerar estruturas, como visto, acarreta a necessidade de forçar a visita a estas estruturas "muito embora algumas destas estruturas nunca sequer venham a ser necessárias para uma necessidade particular": uma das razões pelas quais certas coisas da IA deixam de ter usabilidade, como já verificou Riesbeck. Selecionar ou descobrir componentes de sistemas, em contrapartida, não acarreta esforço extra a este sistema. Introduzimos esta estratégia de seleção em nosso modelo conceitual de I-RBC como um mecanismo útil de aceleração do acesso à informação ou do acesso a casos de uma base de casos.

3.6 Domínios apropriados a I-RBC

O tratamento de resposta a indagação baseado em casos se mostra uma metodologia apropriada em domínios de atividades conhecidos como:

(i) domínios de *fraca teoria* ou pouco formalizados, onde são incertas as relações entre importantes conceitos ("weak theory"); e

(ii) domínios de difícil avaliação/mensuração dada a *natureza qualitativa* das situações.

Entre estes domínios destacam-se, por exemplo:

- *Help desk* (centro de chamadas, *call desk*, centro de assistência técnica, centro de suporte, bureau de serviços, centro de gestão estratégica, ou ainda *hot line*);
- Avaliação ou análise de crédito financeiro;
- Avaliação da qualidade de vida associada a projetos de desenvolvimento social.

São todos domínios computacionalmente desafiantes, não apenas pela ausência de teorias correspondentes como também pela possibilidade de se vir a adotar um tratamento computacional – do tipo *indagação-resposta* – através do emprego da comparação entre situações passadas e situações presentes.

A avaliação da *qualidade de vida*, por exemplo – apesar de relevante para a sociedade – costuma ser feita puramente com base em métodos estatísticos (tal como o método distancial ou genebrino **[SLI 97]**). Desconhece-se qualquer tratamento computacional para esta questão. Em especial, são desconhecidos quaisquer tratamentos computacionais baseados em casos. Tratar grupos de comunidades ou municípios como casos encapsuladores de necessidades sociais (Cestão de necessidades sociais) e encapuladores de índices de satisfação (*índice parcial* para cada sub-área da avaliação, *índice grupal* para cada grupo de necessidade comunitária, e *índice síntese* quer para a qualidade de vida quer para projetos de intervenção em andamento) parece constituir uma tarefa de grande apelo para a indagação-resposta baseada em casos.

Examinamos a adequação da I-RBC particularmente nos dois primeiros domínios, em um esforço para construir cenários de realização informática da abordagem. Resumimos, na seqüência, em que consistem estas áreas onde a aplicabilidade do raciocínio por caso tem sido mais acuradamente examinada em nossa investigação.

3.6.1 *Help desk*

Help desk constitui um *ponto* único de intervenção para solução de problemas enfrentados por usuários – quer se trate de usuários de produtos quer se trate de usuários de serviços; quer se trate de usuários internos a uma organização, quer se trate de usuários externos.

O *help desk*, como visto, costuma ser caracterizado como sendo uma tecnologia de resolução de problemas.

Sob o nosso enfoque, porém, tanto *(i)* as soluções de problemas, quanto *(ii)* as demandas por essas soluções podem ser vistas na ótica de um espaço amplo de *indagações* sobre ajudas e de *respostas* sobre soluções que podem ser armazenadas em uma base de casos. Os resultados concretos deste estudo de aplicabilidade estão relatados nas disssertações *Sistemas Help Desk: Metodologias, Aplicações e Estudo de Caso* **[GOM 98]**, *Uma Arquitetura de Sistemas Inteligentes de Apoio ao Usuário* **[GOR 99]** e *Proposta de um Modelo de Automação do Planejamento da Qualidade através da Utilização de Sistemas de Help Desk que Empregam Raciocínio Baseado em Casos* **[ALB 00]**.

A dissertação de F. L. Gorgônio é particularmente interessante porque explora a questão da indagação-resposta através de uma arquitetura de *help desk* baseada em casos e regras com suporte de Internet. A Figura 3.4 representa este modelo de *Web Help Desk*. A arquitetura, como se vê, compreende uma Base de Conhecimento representado como casos ou como regras, um servidor WWW, uma Equipe de Atendimento e uma Equipe de Consultores no domínio de aplicação. A idéia central na arquitetura é fazer com que somente problemas de extrema complexidade no domínio cheguem às mãos dos consultores.

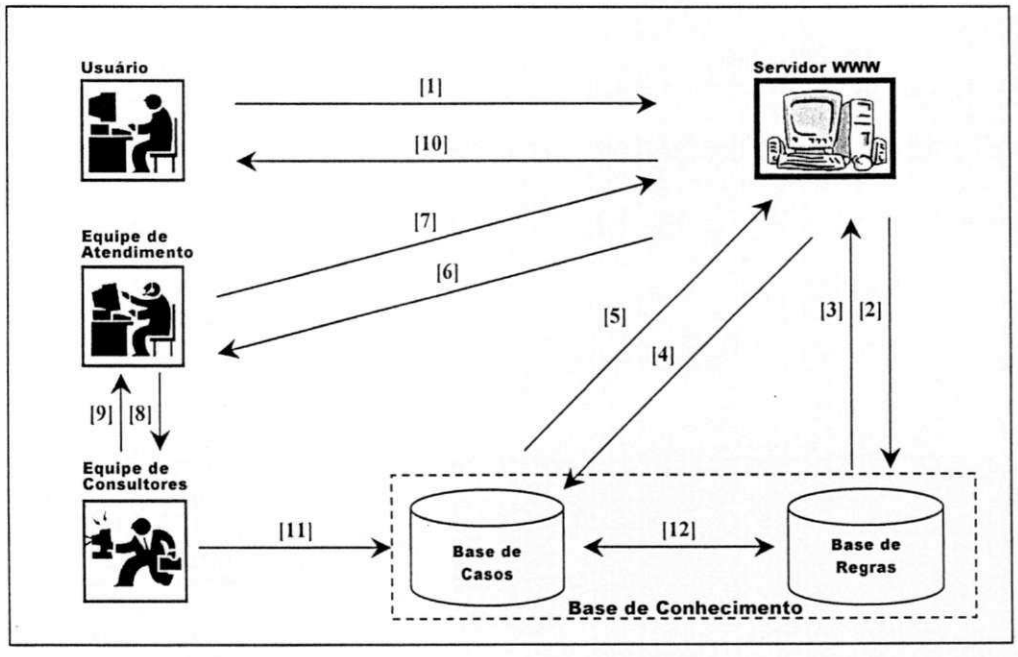

Figura 3.4: O modelo de um sistema do tipo Web Help Desk apoiado em RBC **[GOR 99]**

Ao necessitar de suporte, o usuário se conecta à Internet e acessa o suporte **[1]** através do servidor WWW da empresa. Em uma primeira instância, o sistema irá tentar responder diretamente à indagação do usuário através de busca à base de regras **[2]**, que soluciona problemas que ocorrem com maior freqüência **[3]**. Quando o sistema não consegue uma resposta direta para o problema do usuário, isto significa que, numa primeira instância, não existe qualquer regra na Base de Regras capaz de responder exatamente à indagação feita. O passo seguinte será tentar obter da Base de Casos um

caso que seja semelhante ao problema trazido e cuja solução possa ser utilizada para a situação desse usuário. Assim, o sistema realiza uma busca na Base de Casos **[4]** e recupera aqueles casos que mais se assemelhem ao problema trazido pelo usuário **[5]**. Os casos resgatados são então apresentados ao usuário, dispostos em ordem de semelhança e segundo algum critério de similaridade. Se, porém, o problema trazido também não estiver previsto na base de casos, este problema é repassado para a Equipe de Atendimento **[6]**. Ao resolver rapidamente o problema, uma resposta é enviada **[7,10]** ao usuário. Só então, quando a equipe de atendimento não puder resolver o problema, ela poderá acionar a Equipe de Consultores **[8,9]**. Cabe ainda à Equipe de Consultores a responsabilidade de criação e validação de novos casos a povoarem a Base de Casos **[11]** de modo a permitir a aprendizagem do sistema. Também pode-se prever mecanismos de transformação de casos em regras **[12]**, e vice-versa, que também são formas de aprendizagem de máquina.

3.6.2 Avaliação de crédito financeiro (*loan underwriting*)

Avaliação de crédito bancário é o processo empregado por organismos creditícios públicos ou privados tendo em vista analisar *qualitativamente* solicitações de crédito, ordenar e, por fim, liberar ou não estas solicitações financeiras presentes.

A figura 3.5 ilustra este processo de avaliação e concessão de crédito, um processo dominado por indagações e respostas, tanto da parte do agente financiador quanto do próprio cliente tomador de empréstimo. Na figura em apreço, as letras A, B, C, e D representam o conjunto das solicitações de crédito, em certo momento. O método estatístico do *credit scoring* constitui um dos mecanismos mais usados para avaliação destas solicitações **[LEW 94]**. O método permite estabeler *atributos*, permite estatisticamente estabelecer os seus respectivos *valores* e ainda *valores limiares* para julgar solicitações correntes, com base em solicitações passadas.

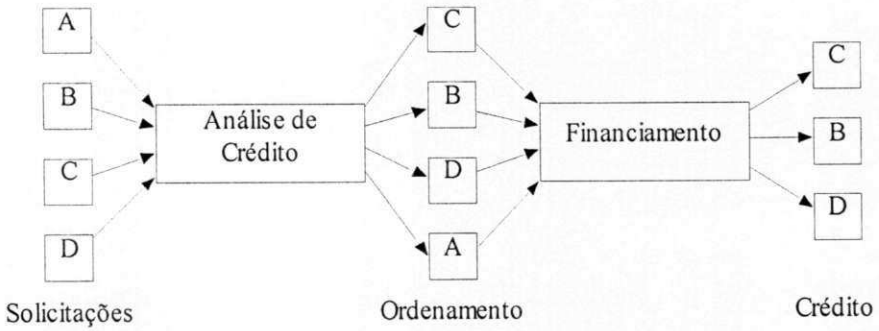

Figura 3.5: Procedimentos básicos da *Análise de Crédito* **[BUA 97]**

Em síntese, o método estatístico do *credit scoring* trata de construir as curvas ilustradas na Figura 3.6 de modo a indicar faixas percentuais dos clientes bons e ruins e os respectivos riscos envolven-

do estes clientes. Estes clientes de empréstimos podem ser classificados, segundo o risco resultante das operações, em cinco faixas como mostradas na figura: risco mínimo = *a*; aceitável = *b*; médio = *c*; considerável = *d*; alto = *e*. Quando dizemos que um cliente é risco *a*, estamos a dizer que a probabilidade desse cliente deixar de pagar um empréstimo será mínima. Já um cliente com risco na faixa *e* oferece uma probabilidade de perda tão alta que não compensa correr o risco de "bancar" um certo empréstimo.

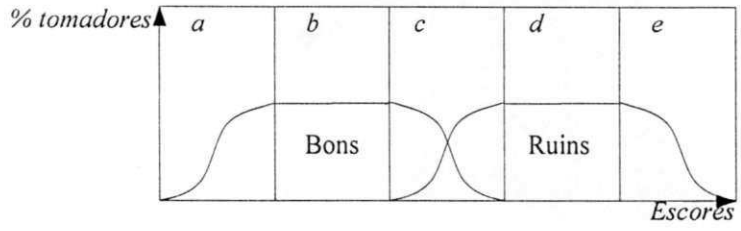

Figura 3.6: Classes de risco do cliente segundo o *credit scoring*

Solicitações de crédito passadas representadas pelas curvas acima permitem, por comparações, responder inúmeras indagações sobre as demandas de crédito do presente. É justamente esta *natureza de caso* subjacente à metodologia estatística do *credit scoring* que nos compele a tratar as decisões sobre crédito através de uma abordagem de RBC tal como no modelo I-RBC aqui analisado. Com base nesta natureza de caso, por nós descoberta, nas tarefas de análise de crédito, os Capítulos 4, 5 e 6 seguintes estabelecem as bases teóricas desta proposta de tratamento de casos para o domínio do crédito. O Capítulo 7 vai tratar da realização experimental deste tratamento incluindo ainda comparações com diferentes metodologias alternativas que têm sido empregadas para este mesmo domínio.

3.7 Conclusão

O presente capítulo ofereceu uma visão geral de nossa investigação. Ele cumpriu uma dupla finalidade: *(i)* desenvolver uma caracterização do paradigma de indagação-resposta em geral (*Query-Answering*); e *(ii)* introduzir, neste contexto, as contribuições metodológicas a serem oportunamente detalhadas nos capítulos posteriores. O resultado deste acoplamento foi a concepção de um *framework* apropriado ao desenvolvimento da indagação-resposta baseada em casos - o nosso modelo I-RBC proposto. Foram destacados no capítulo os conceitos teóricos básicos e as conexões conceituais embasadoras deste *framework* com ênfase inclusive para as condições, os seus componentes, as origens da abordagem, e exemplos de áreas úteis de maior interesse aplicativo.

Capítulo 4

Similaridade e *ranking* de casos baseados na teoria de Tverski-Gati

Pesquisas sobre como as pessoas julgam a
similaridade, aqui, são certamente relevantes.

Janet Kolodner, [**KOL 91**, p. 16]

4.1 Introdução

O presente capítulo trata da construção de um modelo de computação da similaridade entre objetos representados como casos do paradigma de RBC e que estamos a denominar de modelo *SIM(m,p)* de similaridade. Já no Capítulo 2, ofereceu-se uma visão geral dos processos fundamentais que compreendem a tecnologia do RBC e, na seção 2.7.9, foi introduzido o problema da similaridade como um destes processos fundamentais. A abordagem da seção 2.7.9, porém, é apenas descritiva e sem uma maior qualificação da problemática da similaridade. O Capítulo 3, por outro lado, trata da formulação de uma abordagem *de indagação-resposta* onde as respostas computacionais são vistas como *objetos elucidativos* que precisam ter as suas similaridades mensuradas. Neste contexto de indagação-resposta baseado em casos, mostrou-se então o posicionamento do modelo *SIM(m,p)* como sendo um mecanismo desenhado para viabilizar a comparação de respostas armazenadas. O modelo de similaridade foi então apenas introduzido como um dos componentes daquele processo mais amplo de indagação-resposta (como visto na seção 3.5).

A concepção *SIM(m,p)* de similaridade, neste capítulo, parte da idéia de que a similaridade de casos necessita basear-se em teorias para que a sua computação possa inspirar maior confiabilidade. O objetivo então é fazer a descrição detalhada do modelo proposto como também mostrar a base teórica que lhe dá suporte (no caso, a teoria cognitiva de Tversky-Gati sobre a similaridade de objetos).

Os objetivos da investigação, concretamente, são:

- Mostrar conclusões e motivações por nós extraídas do estado da arte da similaridade (seção 4.2);

- Exemplificar a ausência extrema de princípios teóricos em métricas como a *métrica do cosseno*, amplamente usada para a computação de similaridades (seção 4.3);
- Discutir o modelo psicológico de Tversky-Gati (sobre *similaridade cognitiva*) a ser importado para embasar a nossa concepção de similaridade. Três aspectos deste modelo são especialmente analisados: *(i)* a similitude de atributos, propriamente; *(ii)* adissimilaridade de atributos; e *(iii)* a *diagnosticidade* de atributos (seção 4.4).
- Detalhar a introdução de *computabilidade* no modelo psicológico de Tversky necessária para originar e viabilizar a concepção *SIM(m,p)* de similaridade de casos do RBC (seção 4.5).

A adoção da teoria de Tversky – como embasamento para a similaridade necessária ao RBC – tem a vantagem de trazer consigo importantes *insights* sobre o modo humano de perceber similaridades entre objetos. É de se esperar que a credibilidade e as evidências psicológicas dessa teoria já testada através de técnicas próprias da psicologia cognitiva (tais como as técnicas de *estímulos semânticos*, e de *estímulos perceptivos*) possam também se transferir para a sua extensão ao RBC através dos modelos de similaridade e de *ranking* de casos aqui propostos.

4.2 Estado da arte da similaridade: conclusões e motivações

A habilidade de perceber similaridades e analogias constitui um dos aspectos mais fundamentais da cognição humana. Ela é crucial para as atividades de *reconhecimento*, de *classificação*, de *aprendizagem* e desempenha um papel muito importante na *descoberta científica* e na *criatividade*. Daí a relevância crescente desta problemática para as áreas da Inteligência Artificial e das Ciências Cognitivas, com ramificações em diferentes sub-ramos da computação, tais como *(i)* Aprendizagem de Máquina e Reconhecimento de Padrões **[RUS 95, FOR 86]**, *Resgate Inteligente de Informação* **[FOR 86]**, *Query-Answering* **[AND 97]** e Raciocínio Baseado em Casos **[KOL 96]**.

Em relação a nosso domínio particularmente de interesse, a importância da similaridade foi avaliada desde o início, quando Ray Bareiss e James King durante o II Workshop DARPA sobre o RBC asseveravam **[BAR 89]**:

> [*A questão da similaridade*] "*afeta* TODOS *os aspectos do raciocínio baseado em casos*"

Mostramos, a seguir *(i)* a razão de ser desta importância da similaridade; *(ii)* a necessidade de tratamentos mais fundamentados para essa questão; e *(iii)* as motivações para uma concepção alternativa de métrica de similaridade que leve em conta uma base teórica confiável.

4.2.1 Similaridade: alicerce do RBC

A relevância da busca de similaridade decorre de importantes características dos processos do RBC, tais como:

- Similaridades entre atributos salientes de um novo caso e os atributos de casos passados vão *sugerir* aqueles casos relevantes a serem resgatados de uma base de casos.
- Similaridades entre atributos outros de um caso novo e os demais atributos de casos passados poderão *confirmar* essa relevância dos casos resgatados.
- Dissimilaridades envolvendo atributos relevantes de casos já conhecidos poderão *guiar* a adaptação de uma solução passada para uma nova situação.
- Dissimilaridades entre casos que levem ao insucesso de alguma solução (ou a uma solução não apropriada) podem *disparar processos* de aprendizagem que resultem *(i)* ou na retenção de casos; *(ii)* ou no refinamento dos índices; *(iii)* ou mesmo em nova eliciação de conhecimento para modelagem de casos.

4.2.2 Similaridade: necessidade de *princípios*

A computação da similaridade e da dissimilariade dentro do RBC, não obstante a sua enorme importância, continua um problema ainda em aberto (juntamente com questões outras como *matching* seletivo, indexação/seleção de índices, organização da memória, ligações entre casos, memória esparsa, esquecimento de casos, avaliação de casos). *Métricas* para o emparelhamento de casos têm sido, em especial, a origem de muitas dificuldades porque lhes falta aquela fundamentação reclamada nas conclusões do II *Workshop* DARPA sobre o RBC ainda em 1989:

> *"Muitos sistemas existentes empregam esquemas ad hoc para emparelhamentos contra casos na memória (matching).* NECESSITAMOS DESENVOLVER MÉTODOS MAIS FUNDAMENTADOS EM PRINCÍPIOS ("more principled methods"). *Também necessitamos experimentar com métricas para determinar os melhores emparelhamentos de casos".*

O cerne da questão é justamente este:

- Como tem funcionado o uso das métricas de similaridade, tão vitais para a tecnologia de RBC?
- Que *princípios* e onde encontrar estes princípios a fundamentarem a descoberta da similaridade entre casos?

Investigamos estas questões e apontamos a seguir algumas de nossas conclusões motivadoras da

concepção *SIM(m,p)*.

4.2.3 Similaridade: as motivações

O exame do estado da arte da computação da similaridade nos conduz a alguns resultados interessantes:

- Métricas de similaridade parecem ter funcionado tal como produto retirado de alguma prateleira diretamente para usos, sem quaisquer maiores questionamentos: *(i)* sobre as suas *naturezas*, *(ii)* sobre as suas *qualidades* e, sobretudo sem questionamentos, *(iii)* sobre as suas *evidências cognitivas*.
- Métricas, na maioria, apresentam certa elegância matemática ou estatística mas carecem de quaisquer princípios embasadores. Os seus propositores parecem ignorar o fato de que *perceber similaridades fundamentalmente é um ato cognitivo*. Conseqüentemente, o modo humano de perceber e de se comportar perante a comparação de objetos deve ser um dos princípios básicos a nortear a busca e a mensuração da similaridade nos vários ramos da computação onde ela se fizer necessária.
- Métricas, portanto, têm sido desenvolvidas sem incorporar aqueles elementos cognitivos que desde 1989 **[BAR 89]** (*Workshop* de Pensacola Beach) até hoje têm sido reclamados por Janet Kolodner **[KOL 91**, p. 16]: "*Pesquisas sobre como as pessoas julgam a similaridade aqui são certamente relevantes*".
- Métricas convencionais costumam ainda medir a similaridade "pela somatória das similaridades particulares entre pares de atributos de dois casos" **[MAH 95]**. Uma função matemática costuma formalizar essa similaridade entre pares de atributos. Porém, essa mesma função que considera as similaridades de atributos não costuma formalizar, por exemplo, as dissimilaridades destes atributos, como se as diferenças entre atributos em nada possam influenciar a percepção da similaridade, propriamente.

Selecionamos do estado da arte das métricas de similaridade aquela denominada *métrica do cosseno* – como um exemplo bem representativo da abordagem euclidiana da similaridade a merecer as considerações que seguem.

4.3 Similaridade: abordagens euclidianas e do *vizinho mais próximo*

Modelos geométricos têm ultimamente dominado o processamento da similaridade em RBC. Inúmeras métricas representam cada caso como um ponto em algum espaço de coordenadas (em geral, o espaço euclidiano) na esperança de que a distância métrica entre estes pontos possa vir a refletir

as similaridades observadas entre os respectivos objetos ou casos, como na abordagem do RBC (Ver, particularmente, a discussão sobre o assunto feita por I. Watson em [**WAT 97**, p. 25]). Tudo funciona como se este contexto geométrico deva ser a única base a nortear o funcionamento, por exemplo, da *métrica do cosseno* para o cálculo de similaridades que detalhamos na seqüência.

4.3.1 Métrica do cosseno

Suponha que os casos de uma base estejam armazenados na forma de vetores. Suponha ainda que tanto as indagações (Q) quanto os casos (C_j) estejam representados, respectivamente, por conjuntos de indagações e atributos com pesos, como mostrado na Figura 4.1. Nesta figura, s designa a métrica do cosseno que tem sido, popularmente, usada para funcionar como uma função de emparelhamento de vetores objetivando computar o grau de aproximação entre os pares <casos, indagação:>.

Indagação:	$Q = (q_1, q_2, ..., q_t)$, onde q_i é o peso do i-ésimo atributo de Q
Caso j:	$C_j = (c_{j1}, c_{j2}, ..., c_{jt})$, onde c_{ji} é o peso do i-ésimo atributo do caso j
Função de Similaridade: $0 \leq s \leq 1$	$s(Q,C_j) = \dfrac{\sum_{i=1}^{t}(q_i \bullet c_{ji})}{\sqrt{\sum_{i=1}^{t}(q_i)^2 \bullet \sum_{i=1}^{t}(c_{ji})^2}}$

Figura 4.1: Métrica do cosseno para similaridade entre <caso, indagação>

4.3.2 *Ranking* utilizando similaridade do cosseno

Uma vez encontrados, através da fórmula $s(Q, C_j)$, os vários valores das similaridades entre <casos, indagação>, um *ranking* desses casos pode ser estabelecido na ordem decrescente dos valores da função de similaridade, podendo assim esse resultado ser apresentado ao usuário. A Figura 4.2 ilustra esta apresentação de resultados.

Rank dos casos	Identificação dos casos	Coeficiente de similaridade
1	80	0.5084
2	102	0.4418
3	81	0.4212
10	82	0.2843
11	193	0.2771

Figura 4.2: *Ranking* utilizando similaridade do cosseno

Um algoritmo alternativo de *ranking* pode também escolher um certo limiar s' (por exemplo, $s' = 0.75$ para $0 \leq s \leq 1$). Nesta hipótese, todos aqueles casos para os quais as similaridades entre <casos, indagação> venham a resultar em $s \geq s'$ deverão ser resgatados para a atenção do usuário.

4.3.3 Abordagens do tipo *vizinho mais próximo*

Uma outra variante das métricas influenciadas pelo espaço euclidiano são as abordagens estatísticas do tipo *vizinho mais próximo* (também ditas *K-vizinhos mais próximos*), a exemplo dos procedimentos BOB (*Bounds-Overlap-Ball*) e BWB (*Ball-Within-Bounds*) [**ALT 95**, p. 17]. BOB e BWB são algoritmos relativamente simples, porém, puramente geométricos. Eles permitem que o sistema possa testar se um dado nó vem a gerar candidatos para uma lista de *K-vizinhos mais próximos*. O fundamento é o seguinte:

- Uma indagação X_q (um caso-indagação X_q) deve ser considerado como estando no *centro* de uma bola q-dimensional, cujo raio deve ser exatamente providenciado de tal modo que o k-ésimo vizinho esteja localizado na surperfície desta bola. Todos os casos de uma lista de prioridade estão localizados dentro desta bola (cujo tamanho pode ser diretamente definido pelo usuário através de um *limiar*) .
- Um caso pertencendo ao espaço de casos será então um candidato a uma operação de resgate somente se ele for localizado dentro desta bola. Isto é: se a sua similaridade com X_q – medida em distâncias – for maior do que a similaridade entre X_q e o k-ésimo vizinho.

4.3.4 As críticas

Modelos de similaridades como estes têm recebido severas críticas, sobretudo na comunidade de analogia, e as próprias críticas de Tversky e Gati [**TVE 78**] pelo aparato conceitual exclusivamente geométrico. De fato, estas abordagens comprovam as nossas conclusões da seção 4.2.3 sobre as visões simplistas de similaridade. Elas costumam levar em conta apenas fatores quantitativos do tipo *distância*, do tipo *quantidade de atributos*, etc. Ignoram, por conseguinte, toda a riqueza de conhecimentos que as pessoas empregam ao julgar similaridades de situações e de objetos. Ou seja: tais modelos são, na maioria, desprovidos de conhecimentos de *background* tais como *(i)* o conhecimento semântico; *(ii)* o conhecimento sobre o contexto dos objetos em comparação; ou mesmo *(iii)* os componentes cognitivos que devem respaldar a percepção da similaridade [**MAR 99a**].

Por exemplo, se a similaridade entre dois casos (representando situações ou objetos) for definida apenas pela distância no espaço euclidiano, então a distância do objeto A para o objeto B e deste objeto B para o objeto A dentro deste mesmo espaço será necessariamente simétrica. Sob uma tal

abordagem euclidiana, portanto, nunca passará pela mente de quem busca similaridades ou mesmo de um investigador a idéia de procurar *assimetrias* ou *diferenças entre objetos*, como já argumentava Tversky.

4.4 Fundamentos da concepção *SIM(m,p)*: *modelo de contraste*

Modelo de contraste é a denominação dada a uma abordagem de similaridade desenvolvida por Tversky-Gati e que leva em conta assimetrias ou diferenças entre objetos (*the contrast model* [**TVE 78**, p. 80]). Neste sentido, ele se opõe não apenas às métricas puramente geométricas mas também a muitas outras métricas do tipo *vizinho mais próximo*. Tanto as abordagens geométricas e do *vizinho mais próximo* quanto o *modelo de contraste* estão baseados nos atributos dos objetos a serem comparados. Porém, as semelhanças nestas abordagens se encerram por aí.

A teoria de Tversky se torna interessante para o contexto de nossa investigação porque ela incorpora conhecimentos de *background* na mensuração das relações de similaridade de diferentes maneiras. Suas principais características são:

- Cada objeto tverskyano é visto ou caracterizado por um conjunto de atributos ou descritores;
- Associados a cada atributo de certo objeto estão o próprio *valor* do atributo e também a correspondente *diagnosticidade* deste atributo (um conceito que nos será muito útil em nossa extensão do modelo para a arena do RBC);
- A formulação da *função de emparelhamento de atributos*, por fim, faz toda a diferença em relação a métricas convencionais de similaridade. Pela função de Tversky, tanto aqueles atributos que são compartilhados ou comuns aos objetos da comparação quanto aqueles atributos não compartilhados de cada objeto, todos devem contribuir para a mensuração da similaridade entre objetos.

Além dos seus fundamentos cognitivos, portanto, a teoria de Tversky traz a vantagem de elegantemente combinar – em uma única fórmula – tanto as diferenças quanto as similitudes de objetos. Este suporte cognitivo está na base de nossa motivação para estender o modelo teórico de Tversky do domínio da psicologia (onde foi devidamente comprovado) para o domínio do RBC tendo em vista enfrentar o problema da computação de similaridade entre casos, de um modo mais realista. A parte restante desta seção 4.4 detalha essa formulação da similaridade tverskyana. Por outro lado, a parte final deste capítulo (seção 4.5) expõe a sua aplicação na formulação da concepção *SIM(m,p)* de similaridade de casos do RBC.

4.4.1 Relações entre atributos de objetos tverskyanos

Nossa interpretação do *modelo de contraste* inclui quatro aspectos que lhe são fundamentais:

- A importância dos atributos comuns aos objetos bem como dos atributos não compartilhados (nesta seção 4.4.1);
- O conceito de *similaridade cognitiva* expresso através da função de emparelhamento de atributos (*attribute-matching function*, seção 4.4.2);
- A correlação entre similaridades e diferenças de atributos (seção 4.4.3);
- A correlação entre similaridade e a saliência de atributos ou *hipótese da diagnosticidade* de atributos (seção 4.4.4).

Sejam p e i dois objetos em domínios quaisquer. Sejam P e I, similarmente, os conjuntos daqueles atributos que descrevem esses objetos p e i, respectivamente. As relações entre estes objetos estão mostradas na Figura 4.3.

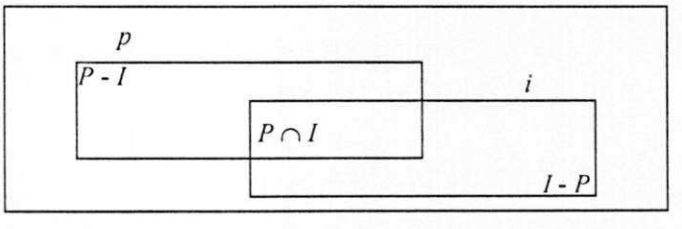

Figura 4.3: Relações entre atributos dos objetos p e i

Três classes de relações entre atributos de objetos estão presentes nesta figura:

- $P \cap I$: ou seja, a relação compreendida pelo conjunto dos atributos compartilhados por ambos os objetos p e i;
- $P - I$: a relação compreendida pelo conjunto dos atributos que estão presentes em p mas que não são compartilhados pelo objeto i; e finalmente
- $I - P$: a relação compreendida pelo conjunto dos atributos do objeto i que não são compartilhados pelo objeto p.

4.4.2 Similaridade cognitiva: conceito

Tversky emprega estas classes de relações entre objetos para definir a sua *similaridade cognitiva* $sim(i,p)$ entre os objetos i e p através da seguinte função de emparelhamento de atributos (*attribute-matching function*):

$$sim(i,p) = \theta f(I \cap P) - \alpha f(P - I) - \beta f(I - P), \text{ onde } \theta, \alpha, \beta \geq 0 \qquad \textbf{[Eq. 1]}$$

Note-se que *f* é uma *função implicitamente definida* como uma combinação linear (ou um *contraste*, de onde vem o nome de *modelo de contraste*) das mensurações tanto dos atributos em comum ou compartilhados quanto dos atributos distintos ou não compartilhados. Naturalmente, isto significa que a similaridade entre dois objetos aumenta na medida em que aumentam as mensurações dos seus atributos comuns aos dois objetos e que também essa similaridade decresce ao tempo em que decrescem as medições dos atributos não compartilhados.

Por outro lado, na Equação 1, os parâmetros $<\theta, \alpha, \beta>$ estão a representar um certo número de pesos ou votos para refletirem a relevância ou proeminência que se queira dar aos atributos comuns ou compartilhados (no caso do parâmetro θ) e a relevância que se queira dar aos atributos não compartilhados (no caso dos parâmetros α e β). Observe-se bem o papel que estes parâmetros estão a desempenhar na definição da similaridade cognitiva. A depender dos valores destes parâmetros, o modelo de contraste em vez de definir um índice único de similaridade, de fato, ele é capaz de definir uma família de similaridades sobre o mesmo conjunto de objetos que estejam em comparação. Considera-se que esta propriedade do modelo de contraste seja de particular importância para efeito de nossa importação para o raciocínio baseado em casos (como ver-se-á na seção 4.5).

Por exemplo, se valores forem estabelecidos tal que $\theta = 1$, quando $\alpha = \beta = 0$, então a similaridade será igual à medida dada pela Equação 2:

$$sim(i,p) = f(I \cap P) \qquad \textbf{[Eq. 2]}$$

Isto é, a similaridade computada será igual à medição encontrada para os atributos compartilhados dos objetos. Por outro lado, se valores forem estabelecidos para $\theta = 0$, e $\alpha = \beta = 1$, vale então a Equação 3:

$$-sim(i,p) = f(P - I) + f(I - P) \qquad \textbf{[Eq. 3]}$$

Isto é, a *dissimilaridade* entre o objeto *i* e o objeto *p* será igual à medição encontrada para a diferença simétrica dos respectivos conjuntos de atributos.

Este é o cerne da teoria cognitiva que estamos a importar para poder dar origem a um mecanismo de similaridade com as características de ser – ao mesmo tempo – *operacional* e *útil* à tecnologia do raciocínio baseado em casos. Outras correlações importantes nesta teoria precisam, no entanto, serem examinadas na seqüência.

4.4.3 Similaridades × diferenças: hipóteses sobre correlações

A hipótese que foi formulada por Tversky consistia em saber se julgamentos de similaridade e julgamentos de diferenças podiam estar perfeitamente correlacionados. Em termos formais, essa hipótese pode ser enunciada do seguinte modo:

- *Hipótese de trabalho*: Faça $sim(a,b)$ e $dif(a,b)$ denotarem, respectivamente, mensurações ordinais de similaridades e de diferenças. De um lado, espera-se que $sim(a,b)$ cresça com $f(A \cap B)$ e decresça com ambos $f(A - B)$ e $f(B - A)$. Contrariamente, espera-se que $dif(a,b)$ decresça quando cresça $f(A \cap B)$ e cresça quando ambos $f(A - B)$ e $f(B - A)$ cresçam.
- *Pressupostos*: *(i)* Suponha que a mensuração da similaridade e a mensuração da diferença entre objetos satisfaçam ao *modelo de contraste*, com sinais contrários mas com pesos diferentes. *(ii)* Suponha ainda que ambas essas mensurações sejam simétricas, por simplificação.

Para testar a *hipótese de trabalho*, aplicando-se o modelo de Tversky, existirão as constantes θ e λ, não negativas, que refletem pesos relativos dados a atributos comuns na percepção de similaridades e diferenças entre objetos e tais que:

$$sim(a,b) > sim(c,e) \Leftrightarrow \theta f(A \cap B) - f(A - B) - f(B - A) > \theta f(C \cap E) - f(C - E) - f(E - C) \quad \textbf{[Eq. 4]}$$

$$dif(a,b) > dif(c,e) \Leftrightarrow f(A - B) + f(B - A) - \lambda f(A \cap B) > f(C - E) + f(E - C) - \lambda f(C \cap E) \quad \textbf{[Eq. 5]}$$

Atribuiu-se aqui o valor 1 aos pesos associados aos atributos não comuns, sem perda de generalidades. Observe-se que, se θ for um valor muito grande, o ordenamento da similaridade vai ser determinado, essencialmente, pelos atributos comuns. Por outro lado, se λ for muito pequeno, o ordenamento das diferenças vai ser determinado pelos atributos não comuns. Conseqüentemente, $sim(a,b) > sim(c,e)$ e $dif(a,b) > dif(c,e)$ podem ser obtidos sempre que:

$$f(A \cap B) > f(C \cap E); \text{ e } f(A - B) + f(B - A) > f(C - E) + f(E - C) \quad \textbf{[Eq. 6]}$$

Conclusão:

se os atributos comuns receberem maiores pesos em julgamentos de similaridades do que em julgamentos de diferenças (como confirmam os experimentos cognitivos),

então um par de objetos – com muitos atributos em comum e também com muitos atributos não comuns – pode ser percebido como sendo *mais similar* e também *mais diferente* do que um outro par de objetos com uma quantidade menor de atributos comuns e não comuns.

4.4.4 A *hipótese da diagnosticidade*

A *saliência de atributos* de objetos constitui um outro importante aspecto da teoria deTversky-Gati [**TVE 78**, p. 92] a merecer consideração, justamente porque julgamos este fator como relevante à extensão da teoria para o domínio do RBC. Na fórmula de Tversky, f geralmente não será invariante em relação a contextos ou a quadros de referência. Isto é, a proeminência de atributos poderá variar de modo amplo, a depender de instruções implícitas ou explícitas ou mesmo a depender do conjunto de objetos que esteja em consideração. Por exemplo, Coréia do Norte e Coréia do Sul podem ser vistas como sendo altamente *similares* do ponto de vista geográfico ou mesmo cultural (geográfico e cultural sendo os atributos proeminentes). Porém, estes mesmos objetos podem ser vistos como sendo bastante *dissimilares*, do ponto de vista do sistema político. Demais a mais, as duas Coréias provavelmente podem ser vistas como sendo *mais similares* entre si, em um contexto que venha a incluir muitos países europeus e americanos do que em um contexto que inclua somente os próprios países asiáticos.

Saliência de atributos é um componente, portanto, a influir na similaridade. Como então a saliência de atributos vem a se alterar à medida em que também mudam os objetos sob consideração?

> *Tversky assevera que essa saliência de atributos é determinada, pelo menos em parte, pela* DIAGNOSTICIDADE *(isto é, a significância classificatória de certo atributo). Um atributo pode adquirir um valor de diagnosticidade (e daí se tornar mais saliente) em um contexto particular se ele vier a servir de base para classificações naquele contexto particular.*

As relações entre similaridade e diagnosticidade formam o que foi denominado de *hipótese da diagnosticidade*, relações estas que estão amplamente comprovadas nos experimentos de Tversky [**TVE 78**, p. 92].

4.5 Uma extensão computacional para o modelo tverskyano

Como então tornar operacional para o RBC esta teoria de similaridade deTversky? Como colocá-la a serviço de nosso modelo I-RBC? Pudemos identificar problemas cruciais a envolver o *modelo de contraste*. Três principais dificuldades precisam ser resolvidas se se pretender conceber uma modelagem computacional para este tipo de similaridade, quais sejam:

- *Problema 1*. O modelo de contraste é um modelo com alto nível de abstração (*feature-theoretical approach*). Ele não foi, originalmente, criado e orientado para a computação e muito menos ainda não foi originalmente orientado para o raciocínio baseado em casos (mesmo porque, na decada de 70, o RBC simplesmente inexistia). O problema fundamental aqui é o de como introduzir *computabilidade* no modelo abstrato de Tversky-Gati.
- *Problema 2*. A função f do modelo é uma função implicitamente definida. Por conse-

guinte, não se pode ver propriamente a "cara" desta função. Ela serve para fornecer *insights* sobre a similaridade abstrata, como visto na seção 4.4, mas uma *f* implícita não facilita a computação desta mesma similaridade.

- *Problema 3.* Uma terceira dificuldade diz respeito à diagnosticidade de atributos. Sabemos, por exemplo, que os parâmetros $<\theta, \alpha, \beta>$ do modelo de contraste denotam pesos ou votos para valorizar ou não ora a interseção ora as diferenças de atributos de objetos. Sabemos, inclusive, a posição destes parâmetros dentro da função *f* de similaridade. A mesma coisa, porém, não fica expícita em relação ao processamento operacional da diagnosticidade.

Estendemos o modelo cognitivo ou *modelo de contraste* nele introduzindo a operacionalização necessária para a computabilidade do modelo. A Figura 4.4 ilustra em que sentido se desenvolve este alargamento computacional para o modelo teórico de Tversky-Gati.

Figura 4.4: Modelo computacional como extensão do *modelo cognitivo*

Observe-se na figura que os objetos abstratos de Tversky são computacionalmente tratados sob quatro aspectos, a exigirem diferentes mecanismos metodológicos: *(i)* o tratamento metodológico para indexação de casos; *(ii)* o tratamento metodológico para a similaridade cognitiva; *(iii)* o tratamento metodológico para o *ranking* de casos resgatados; e *(iv)* o tratamento metodológico para avaliação de bases de casos.

Ou seja, o modelo computacional mostrado na figura tem os seguintes componentes:

- *Tratamento metodológico para indexação.* Objetos tverskyanos, em nossa abordagem, são modelados como *casos* do RBC. Casos, por sua vez, são criados e representados em forma de tabela (semelhante a *frames*) e constituem a primeira solução para a extensão do modelo tomado como base da investigação.

- *Tratamento metodológico para operacionalização da similaridade cognitiva.* A explici-

tação da função de Tversky de uma forma apropriada à computação da similaridade de casos dá origem ao modelo *SIM(m,p)* e constitui a segunda solução para a operacionalização da teoria tverskyana.

- *Tratamento metodológico para ranking de casos*. Estebeleceu-se também que o processo de *ranking* de casos também seja originado, como subproduto, do processo de similaridade cognitiva.
- *Tratamento metodológico para avaliação de base de casos*. Casos criados, representados e armazenados são passíveis de avaliação qualitativa ao serem resgatados pelo usuário.

Para este modelo estendido por nós introduzido, o componente metodológico de criação e de indexação de casos está detalhado no Capítulo 5 seguinte; enquanto isto, o seu componente de avaliação qualitativa de casos que funcionem como respostas ao usuário está exposto no Capítulo 6. Na seqüência, mostramos o tratamento da operacionalização da similaridade e as particularidades dele decorrentes em termos de *ranking* de casos.

4.5.1 Um tratamento operacional para a similaridade tverskyana

A nossa extensão do modelo de Tversky procura inicialmente dar conta de viabilizar a computação da similaridade acima descrita em nível teórico [**MAR 99a, MAR 99b**]. Ou seja, o tratamento operacional da similaridade cognitiva vai exigir uma concepção de casos e um correspondente esquema de representação computacional que leva em conta os seguintes elementos:

- o espaço de atributos de casos;
- os valores dos atributos de casos;
- os pesos ou votos para valores de atributos de casos;
- os pesos ou votos para a diagnosticidade dos atributos; e, sobretudo, apoia-se na versão derivada da
- a função *f* de similaridade cognitiva.

Essa versão operacional da função de similaridade, portanto, vai girar em torno da manipulação desses valores de atributos e de seus respectivos pesos ou votos a eles atribuídos. Vamos estabelecer que o mecanismo para explicitar ou introduzir o parâmetro da *diagnosticidade* de atributos na própria função de similaridade venha a ser o mesmo mecanismo de atribuição de pesos, já existente na literatura. Nesta situação, o peso indicador da *diagnosticidade* vai ficar associado não a valores de atributo (como ocorre normalmente na literatura) mas ao próprio atributo nele mesmo, com o fim de expressar a sua relevância ou a sua essencialidade na definição de objetos e situações.

Isto posto, a métrica de similaridade derivada do modelo tverskyano original que se está a buscar poderá ser construída a partir do estabelecimento dos seguintes parâmetros e variáveis:

(i) m e p: Vão designar, no design da nossa métrica, respectivamente, qualquer caso armazenado (*caso-resposta*), e qualquer caso alvo (*caso-indagação*). São evidentemente casos estruturados, de acordo com a metodologia que será descrita no Capítulo 5 seguinte;

(ii) M: Designa o conjunto dos atributos constituintes e caracterizadores do caso-resposta m;

(iii) P: Designa o conjunto dos atributos constituintes e caracterizadores do caso-indagação p;

(iv) $M \cap P$: Representa aquele conjunto de atributos que estejam na interseção ou atributos compartilhados pelos casos m e p, de acordo com a Figura 4.3 do modelo original;

(v) $M - P$: Vai designar aquele conjunto dos atributos do caso m não compartilhados com o caso p;

(vi) $P - M$: Vai designar, contrariamente, aquele conjunto de atributos do caso p não compartilhados com o caso m;

(vii) D = *diagnosticidade-atributo*: Este parâmetro D vai designar aquele *peso* a ser associado ao atributo de um caso (não ao seu valor, propriamente), de modo a representar o conceito tverskyano de diagnosticidade ou a importância de um atributo enquanto definidor daquele objeto sendo representado por um caso computacional. Ou seja, este parâmetro vai expressar a contribuição de cada atributo na representação de um certo objeto. Suponha o objeto *empréstimo-bancário*, com o qual trabalhamos nas experimentações. Vários atributos vão ser necessários para definir (em tempo de análise de crédito) a existência ou não deste objeto. Para a existência real de uma operação de crédito, a diagnosticidade do atributo *rendimento* certamente será maior do que a diagnosticidade do atributo *profissão*, por exemplo.

(viii) V = votos-*valor-atributo*: Vai designar a quantidade de pesos ou votos dados a uma certa instância do valor de um atributo pelo *designer* do sistema. Por exemplo: o atributo *rendimento*, no mesmo contexto das experimentações com crédito financeiro, pode assumir diferentes instâncias de valores. A cada uma dessas instâncias pode ser atribuído um diferente peso a ser empregado, posteriormente, na computação da similaridade entre casos de crédito (pessoal ou empresarial).

(ix) X = (*diagnosticidade-atributo*) × (*votos-valor-atributo*): Essa expressão está a significar uma ponderação, via diagnosticidade, aplicada sobre os votos dados às instâncias de valores de atributo. No cálculo da similaridade mais à frente, a expressão será aplicada

sempre que conjuntos de atributos estejam presentes em um dos casos mas não no outro caso. Ou seja: sempre que não sejam atributos compartilhados pelos casos *m* e *p*.

(x) $X' = diagnosticidade\text{-}atributo \times Fator \times min(votos\text{-}valor\text{-}atributo(m, p))$: X' será determinada na situação contrária à de X. A variavel X' será determinada quando na hipótese de atributos que estejam presentes tanto no caso *m* quanto no caso *p*, mas, com diferentes pesos pra seus valores (em razão mesmo dessa diferença de valores). Considera-se então o mínimo desses pesos ou votos que tenham sido atribuídos pelo designer. Acha-se X', em seguida, multiplicando-se esta quantidade mínima de pesos por um certo fator de correção, como definido na seqüência.

(xi) *Fator*: Este parâmetro designa um fator de correção expressando uma distância entre a quantidade máxima e a quantidade mínima de pesos dados a valores diferentes de um mesmo atributo, presente no caso *m* e no caso *p*, respectivamente. Esta quantidade *mínima* tomada em X' representa, por conseguinte, uma interseção daqueles pesos em apreço, quantidade essa a ser ponderada por seu respectivo *Fator*, ilustrado na tabela abaixo.

Pesos	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**
1	1	0.88	0.76	0.64	0.52	0.40	0.28	0.16
2	0.88	1	0.88	0.76	0.64	0.52	0.40	0.28
3	0.76	0.88	1	0.88	0.76	0.64	0.52	0.40
4	0.64	0.76	0.88	1	0.88	0.76	0.64	0.52
5	0.52	0.64	0.76	0.88	1	0.88	0.76	0.64
6	0.40	0.52	0.64	0.76	0.88	1	0.88	0.76
7	0.28	0.40	0.52	0.64	0.76	0.88	1	0.88
8	0.16	0.28	0.40	0.52	0.64	0.76	0.88	1

Figura 4.5: Fatores de ponderação sobre pesos mínimos

(xii) $w = (X \vee X')$

(xiii) *SIM(m,p)*: Finalmente, *SIM* vai designar a métrica derivada - a partir da similaridade entre objetos teóricos tverskyanos - para computar a similaridade observada do caso *m* em relação ao caso *p*, tal que:

$$SIM(m,p) = a \sum_{i \in M \cap P} w_i - b \sum_{j \in P-M} w_j - c \sum_{k \in M-P} w_k$$

sendo $a, b, c \geq 0$.

razão desta exemplificação ter sido deixada para o Capítulo 7 sobre as experimentações.

4.5.2 Características do tratamento operacional *SIM(m,p)*

Nesta versão amplificadora e explicitatória da função tverskyana de similaridade, três características básicas precisam ser observadas:

- *Papel dos parâmetros*. Os parâmetros <*a*, *b*, *c*> passam a desempenhar, *em relação a casos do RBC*, papéis correspondentes aos papéis dos parâmetros <θ, α, β>, *em relação aos objetos abstratos* de Tversky: *a* designa os pesos dados aos atributos da interseção entre o caso *m* e o caso *p* (ou atributos compartilhados); enquanto *b* e *c*, por outro lado, designam os pesos dados aos atributos que não estejam na interseção dos casos *m* e *p*;
- *Uniformidade de tratamento*. A computação da diagnosticidade ou do poder classificatório de um atributo está sendo feita também mediante a atribuição de pesos e já vem embutida na computação dos valores para a variável *w*. Tratar a *dignosticidade* de atributo do mesmo modo como são tratados os atributos comuns e não comuns aos casos (isto é, através da mensuração via pesos) garante uma uniformidade de tratamento na computação de *SIM(m,p)*.
- *Pré-processamento da similaridade*. A formulação de *SIM(m,p)* traz um aspecto de flexibilização que permite ao usuário e/ou ao *designer* de sistema intervir nos resultados da similaridade através dos parâmetros <*a*, *b*, *c*>. Todos os cálculos anteriores à atribuição de valores a este parâmetros podem ser vistos como uma espécie de *pré-processamento da similaridade* e vamos supor esse *pré-processamento* ao se discutir a correspondente metodologia de *ranking* de casos, na seção 4.5.3 seguinte.

4.5.3 Um tratamento operacional para o *ranking* de casos: Modelo *ORDEN*

Uma conseqüência importante nesta extensão do modelo cognitivo ao RBC é o seu impacto como algoritmo de *ranking* de casos. Na seção 4.3.2, vimos que os dois mecanismos usualmente empregados no *ranking* de casos eram: *(i)* a ordenação simples das similaridades obtidas por ordem decrescente de seus valores, e a *(ii)* imposição de um certo *limiar* às similaridades obtidas; onde somente valores de similaridade acima deste limiar devem ser exibidos ao usuário. Nosso modelo estendido introduz um mecanismo novo de *ranking* de casos, além dos mecanismos *(i)* e *(ii)* já discutidos. Como o modelo cognitivo original permite que uma *família de similaridades* seja gerada sobre os mesmos objetos que estejam em comparação, isto implica que nosso modelo estendido também permitirá uma família de ordenamentos de casos, segundo suas similaridades e, conseqüentemente, novas formas de visualizações dos casos similares ao Caso X em mãos do usuário.

4.5.3.1 *Ranking* com parâmetros iniciais: *rank-1*

As Figuras 4.6 e 4.7 ilustram este novo mecanismo de *ranking*. Nas colunas da Figura 4.6 tem-se o seguinte: *(i)* identificação dos casos resgatados; *(ii)* pré-processamento das similaridades segundo a fórmula da similaridade *SIM*(*m,p*); *(iii)* primeiro grupo de valores para os parâmetros <a, b, c> necessários à fórmula *SIM*(*m,p*); *(iv)* as similaridades, propriamente, obtidas mediante a substituição destes primeiros valores de parâmetros nas fórmulas pré-processadas; e, finalmente, *(v)* o *rank-1* dos casos, por ordem de grandeza de suas similaridades com o Caso X de entrada. Observe-se, na coluna do *rank-1*, que *SIM*(Caso 1, Caso X) apresenta a menor similaridade encontrada, sendo portanto o Caso 1 o último no *rank* de visualização, ao contrário do Caso 2 que apresenta a maior similaridade.

Identificação dos resgates	Pré-processamento de *SIM*(*m,p*)	Parâmetros iniciais	*SIM*(*m,p*)	Rank-1
Caso 1	a×37 - b×10 - c×4	a = 3 b = c = 1	97	5
Caso 2	a×37 - b×2 - c×4		105	1
Caso 3	a×35 - b×4 - c×3		98	4
Caso 4	a×35 - b×5 - c×5		101	3
Caso 5	a×36 - b×3 - c×2		103	2

Figura 4.6: *Ranking* de casos com parâmetros <*a, b, c*> iniciais

Identificação dos resgates	Pré-processamento de *SIM*(*m,p*)	Parâmetro modificado	*SIM*(*m,p*)	Rank-2
Caso 1	a×37 - b×10 - c×4	a = 4 b = c = 1	134	3
Caso 2	a×37 - b×2 - c×4		142	1
Caso 3	a×35 - b×4 - c×3		133	4
Caso 4	a×35 - b×5 - c×5		130	5
Caso 5	a×36 - b×3 - c×2		139	2

Figura 4.7: *Ranking* de casos com parâmetro *a* modificado

4.5.3.2 *Ranking* com parâmetros modificados: *rank-2*

Suponha agora que o usuário decida modificar a ênfase inicial dada à interseção dos atributos dos casos, ou seja, decida aumentar ainda mais o valor do parâmetro *a*. A Figura 4.7 ilustra esta nova situação, mostrando as novas similaridades decorrentes desta alteração e o *rank-2* dos casos a serem visualizados. Semelhantemente, os parâmetros *b* e *c* podem convenientemente serem alterados.

A metodologia de *ranking*, portanto, é suficientemente flexível para permitir ao usuário modificar o ordenamento dos casos resgatados conforme a sua decisão de valorizar ora a interseção dos atributos dos casos ora as diferenças destes mesmos atributos, nos termos descritos na seção 4.4.3.

4.6 Conclusão

Analisou-se, neste capítulo, o papel da similaridade para o raciocínio baseado em casos, como ainda a necessidade de se dispor de metodologias embasadas em princípios e teorias sobre a percepção da similaridade entre casos. Uma destas teorias da área da psicologia cognitiva – a teoria da similaridade de Tversky-Gati – foi suficientemente estudada no capítulo sob o prisma daqueles aspectos de possíveis relevâncias para o RBC. Uma vez identificada a importância desta teoria cognitiva para os fenômenos da similaridade, mostrou-se como estender essa abordagem cognitiva de tal modo a viabilizar uma metodologia para o RBC que seja, ao mesmo tempo, fundada em princípios e suficientemente operacional na tarefa de comparar casos a serem resgatados de uma base de conhecimento.

A metodologia $SIM(m,p)$ resultante da importação da teoria de Tversky para o RBC também traz para o RBC uma nova possibilidade como mecanismo de *ranking* de casos, conforme foi mostrado ao longo da discussão.

Capítulo 5

Indexação baseada em tabela em suporte à similaridade

"Com o meu sistema paralelo, não se faz necessária qualquer *indexação* [...], muito embora nós ainda necessitemos de uma *teoria* sobre como atribuir valores de importância e sobre como permitir que estes valores possam alterar-se de acordo com o contexto".

(Citado em [**KOL 96**, p. 352], sem mencionar o autor)

5.1 Introdução

O presente capítulo, assim como o capítulo anterior, se enquadra dentro do contexto mais amplo da formulação de um modelo computacional que estenda o modelo cognitivo de Tversky-Gati, de modo a introduzir neste modelo tverskyano original a operacionalização dos seus mecanismos de similaridade. Descrevemos no capítulo anterior, particularmente, as características e aexplicitação formal do modelo *SIM(m,p)* no qual se impõe que casos do RBC passem a representar os objetos do modelo cognitivo – os objetos tverskyanos – e, como conseqüência, esses casos possam ter as suas similaridades computadas tais como objetos tverskyanos. Isto é, possam ser computadas levando em conta: *(i)* espaços de atributos; *(ii)* valores para atributos; *(iii)* pesos para valores; *(iv) diagnosticidade* de atributo. Como então representar computacionalmente casos exibidores de tais propriedades?

Interessa-nos, neste ponto, tratar de explicitar a própria representação dos casos do modelo *SIM(m,p)* do Capítulo 4. Ora, o fato de ter de explicitar a forma de representação de casos remete a nossa investigação para o que existe de mais problemático em RBC – o *problema fundamental da indexação* [**KOL 96**]. Nossa questão, portanto, se traduz nas indagações básicas seguintes:

- Como indexar/representar casos projetados para simularem os objetos tverskyanos anteriormente descritos?
- Como usar esta representação de casos para pré-processar e computar, operacionalmente, a similaridade *SIM(m,p)* entre casos do RBC?

Este capítulo está organizado da maneira seguinte. A seção 5.2 posiciona a problemática da indexação de casos em geral, após uma primeira introdução dessa questão ainda na seção 2.5. Em seguida, na seção 5.3 dá-se um passo à frente na elucidação teórica de conceitos de indexação considerados essenciais mas que, dentro da comunidade de RBC, são conceitos que estão na origem de vasta gama de entendimentos errôneos. A seção 5.4, finalmente, trata de especificar o modelo de indexação e de representação capaz de viabilizar experimentações concretas com o modelo de similaridade *SIM(m,p)*.

5.2 Problema fundamental da indexação

Indexação é uma operação tratada na literatura como tendendo a ser ou tudo ou nada, tamanha é a sua complexidade em RBC: "tende a ser ou o esfoço principal para uns ou quase nada para outros", na observação de Christopher Riesbeck **[RIE 96]**. Ou seja, os relatórios de pesquisas e sistemas de RBC ora demonstram compreender e enquadrar a indexação no algoritmo geral do RBC ora demonstram total ignorância do seu papel, como comprovou J. Kolodner **[KOL 96]**. Toda uma variedade de interpretações errôneas estão a confundir a indexação de casos, neste estágio presente da evolução do RBC.

5.2.1 Concepções errôneas sobre indexação

Interpretações erradas sobre a indexação e o RBC preocupam J. Kolodner desde 1996, ao preparar o seu texto denominado: "*Tornando explícito aquilo que está implícito: clarificando os princípios do raciocínio baseado em casos*" **[KOL 96]**. Neste documento, Kolodner chega a identificar e a criticar diversas posições teóricas, ao mesmo tempo em que – ela mesma – também cria muitas outras dificuldades conceituais. Apontemos os seguintes problemas criados pela autora e por outros:

(i) "Indexação, *talvez* seja um nome errado para denominar o problema da acessibilidade [...]. Acessibilidade depende de *(i) situation assessment*; *(ii)* reconhecimento daquilo que seja importante para a representação; *(iii)* funções de busca; *(iv)* métricas de similaridade; *(v)* vocabulário; *(vi)* funções de emparelhamento" (palavras de Kolodner) .

(ii) "Resgate vem primeiro (entre os processos do RBC) e cujo passo inicial é o de *situation assessment* [...]. Isto leva ao processo de busca [...]" (palavras de Kolodner).

(iii) "Um índice em uma biblioteca de casos é [...] um ponteiro para um caso" (palavras de Kolodner).

(iv) "Com o meu sistema paralelo, *não se faz necessária qualquer indexação* [...], muito

embora nós ainda necessitemos de uma teoria sobre como atribuir valores de importância e sobre como permitir que estes valores possam alterar-se de acordo com o contexto" (Autor omitido por Kolodner).

(v) "Em meu sistema, nós usamos probabilidades condicionais para determinar aquilo a resgatar. Com o meu sistema paralelo que toma decisões usando probabilidades condicionais, não se faz necessária qualquer indexação [...]. Certamente, existe ainda a necessidade de pesquisas para descobrir como bem atribuir probabilidades condicionais" (Autor omitido por Kolodner).

(vi) "Todos nós tivemos experiências com sistemas baseados em casos que, ao possuirem bibliotecas de casos pequenas, têm também um resgate muito rápido. Quando, porém, cresce essa biblioteca de casos, o resgate se torna tão vagaroso que o raciocínio baseado em casos não se torna mais viável" (Autor omitido por Kolodner).

5.2.2 Crítica das concepções

Todas as proposições acima claramente se referem à indexação, com exceção da última. E todas também são criticáveis conforme segue:

- *Proposições 1-2*. Estas proposições, por exemplo, demonstram as inconsistências próprias do estilo de metodologia de J. Kolodner. A autora chega a desconfiar de que indexação seja uma denominação imprópria em RBC **[KOL 96**, p. 355] mas ela não consegue ver que essa impropriedade da denominação decorre do fato de não conseguir estabelecer a correta correlação entre os conceitos de *indexação* e *resgate*. Para Kolodener, quase tudo é resgate e ao mesmo tempo quase tudo é indexação, inclusive a operação de *situation assessment*. "Resgate vem primeiro", conforme afirma a própria Kolodner, no artigo em foco. Ora, para nós, resgate só será possível após a indexação, tal como ocorre em *Information Retrieval* (IR) **[KOW 97]**. Ou seja, após a interpretação, a extração e representação de informação pelo indexador.

- *Proposição 3*. A idéia de índice como *ponteiro* também é uma das fontes de problemas em RBC porque desfigura o conteúdo da operação de indexação. Kolodner identifica esta impropriedade, faz a crítica de quem entende a indexação sobretudo como emprego de ponteiros. A autora, porém, de nenhuma forma soluciona o banimento total desta idéia problemática.

- *Proposições 4-5*. Os autores destas proposições, por exemplo, ignoram o real sentido de indexação para entendê-la como o emprego de ponteiros, nos termos da crítica de Kolodner.

- *Proposição 6*. Na última das afirmações, o autor (não identificado por Kolodner) comenta erroneamente sobre aquilo que torna útil um sistema baseado em casos, esquecendo-se, porém, de que – para muitas aplicações – pequenas bibliotecas de casos são qualitativamente suficientes.

5.3 O que significa indexação? Uma contribuição clarificadora

As problemáticas idéias envolvendo índice e indexação, em geral, são idéias importadas da área de banco de dados para o RBC em virtude da proximidade paradigmática entre bases de dados e bases de casos. Em oposição a estas idéias, nós preferimos clarificar estes conceitos básicos do RBC a partir do ponto de vista da *Information Retrieval* – uma sugestão feita, aliás, pela própria Kolodner [**KOL 96**, p. 358]. Uma visão dos conceitos básicos do RBC à luz da teoria de *Information Retrieval* poderá constituir um *framework* sólido para uma correta interpretação da indexação, dos índices, e do próprio conceito de casos, viabilizando, deste modo, uma contribuição efetivamente clarificativa para o RBC capaz de banir muitos dos atuais *conflitos conceituais* herdados de banco de dados.

5.3.1 Indexação, índice e caso como informação indexada

Podemos formular então o seguinte conjunto de conceituações a partir das mais genéricas para as mais específicas (seguindo-se um estilo *top-down* de análise) [**MAR 99c, KOW 97**].

Definição 5.1: *Indexação de casos*. Indexação é o processo que consiste em analisar uma situação (não necessariamente uma situação textual ou impressa) tendo em vista dela extrair aquela informação relevante a ser permanentemente guardada em um *índice*.

Observe-se a parte final deste conceito: ([...] *a ser permanentemente guardada em um índice*). É justamente este conceito de índice como REPOSITÓRIOS PARA GUARDAR INFORMAÇÃO que argumentamos constituir a própria essência do conceito de caso. *Indexação*, por conseguinte, significa criar esses repositórios – ou casos do RBC – através do seu preenchimento com informações originadas de um processo de análise de situações. Argumentamos ser tal conceito baseado na área de *Information Retrieval* tudo o que deve caracterizar a indexação e os próprios casos do RBC.

Tal conceito de indexação, como se vê, não faz qualquer referência a *processos de resgate*, contrariamente às inconsistências conceituais comuns, por exemplo, em Kolodner [**KOL 96**]. Três são as conseqüências lógicas deste conceito:

- *Indexação* fica caracterizada, fundamentalmente, como um processo de *eliciação de casos* (um tipo particular de eliciação de conhecimento);

- A completa separação entre a indexação e o resgate de casos: uma coisa será criar – via indexação – aqueles casos necessários a uma base de conhecimento; uma outra coisa diferente será armazená-los, e resgatá-los oportunamente;
- *Índice* passa a corresponder para nós ao próprio conceito de caso, uma vez ser um caso uma informação indexada, por analogia com os *Index Databases* que são bancos de dados formados por índices, em IR [**FOX 93**, **KOW 97**, p. 17]. Ou seja – em oposição ao significado de "ponteiro" – índices passam a corresponder a estruturas de casos capazes de guardar aquelas informações extraídas de situações.

Exemplos de índices e de indexação são apresentados mais à frente, após a clarificação dos demais conceitos. Como conseqüência da Definição 5.1, valem também as distinções que seguem.

Definição 5.2: *Índice como estrutura de dados*. Índice é uma estrutura de dados pesquisável (ou navegável), criada para dar apoio a uma estratégia de busca.

Este conceito de índice [**FOX 93**; **KOW 97**, p. 96] que aparece na enunciação do conceito de indexação se opõe totalmente à idéia erroneamente estabelecida em RBC de índice como *ponteiro* – um problema identificado mas conceitualmente não resolvido por Kolodner. Índice, por conseguinte, não é um ponteiro (uma *seedword,* um *termo* no "pronto" de um sistema, como ficou difundido). Índice é a própria informação indexada, quer numa forma livre ou sem a estruturação característica de um caso (como na indexação automática, seção 5.3.1.3) quer já organizado na forma de casos. Em ambas as situações, por definição, um índice é uma estrutura de dado pesquisável. Argumentamos ser justamente este aspecto do novo conceito de índice que deve ser abarcado pelo conceito de caso, tal como apresentado nas discussões seguintes.

Definição 5.3: *Caso como índice*. Caso do RBC é uma estrutura de dado pesquisável (um índice formado por conhecimentos situacionais) criada como um produto de um processo de indexação para dar apoio a uma estratégia de busca.

Definindo-se um caso do RBC em função do conceito de índice – no sentido da Definição 5.2 – e definindo-se a indexação como sendo o processo de criação destes índices compostos por *termos* descritores de atributos situacionais, podemos garantir que não mais o conceito de índice poderá ser confundido com o conceito de ponteiro. Não fará sentido, pois, se dizer que uma estrutura de dados – tal como um índice – seja um ponteiro, muito embora se possa afirmar que um certo *termo* (uma *seedword*) possa apontar para um índice, significando apontar para uma *informação indexada* ou para um *caso*. Termos e conceitos extraídos de situações para comporem esses casos ou informações indexadas podem ser de dois tipos principais: *(i)* atributos que descrevam problemas; e *(ii)*

termos que descrevam soluções. Neste sentido, a nossa proposta de clarificação do conceito de indexação introduz a subsunção (o abarcamento) do conceito de *índice* pelo próprio conceito de caso, não apenas considerando-se a natureza dos processos envolvidos, mas ainda como um mecanismo para banir a idéia de ponteiro associada à indexação.

5.3.1.1 Eliciação de casos e representação de vocabulário

A interpretação correta da Definição 5.1 nos permite fazer ainda os seguintes desdobramentos conceituais.

Definição 5.4: *Indexação como eliciação*. Uma indexação é composta pelas etapas de eliciação de casos e de representação de vocabulário.

Definição 5.5: *Eliciação como "Situation Assessment"*. Eliciação de casos é a operação de RBC constituída pelas etapas de *(i) Situation Assessment* (análise de situações) e de *(ii)* tradução.

Eliciar – de onde se origina o nome *eliciação* – significa fazer sair; fazer vir à tona aquele conhecimento formador de casos (*to elicit knowledge*). O primeiro passo da indexação é a eliciação de casos compreendida pela *(i) análise de situações* e pela *(ii) tradução*. A análise de situação implica decidir sobre aquilo de que trata uma dada situação – fonte de possíveis casos. Por exemplo, uma situação de liberação ou não de um empréstimo bancário ou uma situação de assistência ou não na correção de *bugs* de programas, em sistemas *help desk*. O indexador tem de formular e responder perguntas tais como: *(i)* De que trata uma tal situação? *(ii)* Por que tal situação deve dar origem a casos? *(iii)* Quais os aspectos de uma particular situação que serão de interesse para os usuários de uma base de casos? Quantos casos podem ser originados de uma única situação?

Tradução, ainda em relação à definição de eliciação, é a etapa da indexação que envolve propriamente a conversão (a passagem) da análise de situação para um determinado conjunto de *termos* a comporem casos em criação.

Representação de vocabulário (na Definição 5.4) é o segundo passo da indexação, correspondente à etapa final, que consiste de duas preocupações básicas: *(i)* a representação, *de forma não ambígua*, dos termos da indexação para efeito da modelagem dos casos, incluíndo a preocupação com a combinação destes termos entre si; e *(ii)* a atribuição de *valores de importância* aos termos descritores de atributos de casos. Por conseguinte, atribuir valores de importância aos termos de uma indexação, ou mesmo associar a eles probabilidades condicionais, são esquemas representacionais para se lidar com o problema da indexação – coisa ignorada, por exemplo, pelos autores das *proposições (iv)* e *(v)* da seção 5.2.1.

5.3.1.2 *Situation Assessment*: exemplo

Feitas as diferenciações estabelecidas acima, exemplificamos nesta seção o conceito de *situation assessment*, básico para a indexação. Suponha que se queira analisar situações (ou realizar *Situation Assessment*) orientadas para a modelagem de possíveis casos do RBC úteis para *help desk* em programação.

Suponha uma situação de *help desk* para depuração de programas no contexto do modelo *Web Help Desk* da Figura 3.4. Tanto o programador 1 como o programador 2 estão a depurar programas imperativos. O programador 1 tenta localizar erros sintáticos em seu programa Modula-2, enquanto o programador 2 tenta refazer um *loop* em seu programa Pascal. Quais os *termos* a serem selecionados para representar atributos desta situação? A análise de uma situação concreta como esta dá origem a um conjunto de termos que podem ser dela extraídos ou a ela atribuídos. Mas a indexação vai além. Ela providencia também a representação e o *link* entre si daqueles termos criados para comporem possíveis índices/casos, como mostram as estruturas de (1) a (4) da Figura 5.1.

Termos Atribuídos à Situação	Metodologia da *link*
(1) erro, sintático, programa1, loop, refazer, depuração, programa2, programador 1, Modula-2, programador2, Pascal	Sem *link* de termos
(2) (erro sintático, programa1, depuração, programador1) (loop, programa2, refazimento, programador2)	*Link* (Pré-coordenação)
(3) (programador1, depuração, erro sintático, programa1) (programador2, refazimento, loop, programa2)	*Link* com posições indicando papéis dos termos
(4) (*Sujeito*: programador1 *Ação*: depuração *Objeto*: erro sintático, programa1 *Modificador*: em Modula-2) (*Sujeito*: programador2 *Ação*: refazimento *Objeto*: loop, programa2 *Modificador*: em Pascal)	*Link* (Pré-coordenação) com *modificadores* indicando papéis

Figura 5.1: Indexação por *termos* e seus *links* ou ligações

Os fatos significativos da eliciação de casos mostrados na Figura 5.1 são os seguintes:

- Os termos da estrutura (1) não exibem qualquer *link* entre si. São, simplesmente, termos selecionados para descrever a situação de depuração de programas, do exemplo.
- Nas estruturas (2) a (4) da figura, os mesmos termos estão ligados entre si de três maneiras diferentes. O *link* dos termos na estrutura (2), por exemplo, permite que o sistema não venha a correlacionar erroneamente os *bugs* em Modula-2 e em Pascal, uma vez que estes *bugs* não estão a se localizar nos mesmos vetores de termos que estejam ligados. Tem-se aqui uma ligação ("*linkagem*") de termos denominada de *pré-coordenação*, por ser realiza-

da em tempo de indexação; e não em tempo de busca.

- A estrutura (3), por sua vez, leva em conta o *papel posicional* daqueles termos a serem dispostos em vetores, permitindo um único valor por posição. Que acontecerá então quando mais de um termo de uma situação vierem a concorrer por uma mesma posição, em um certo vetor?

A estrutura (4) constitui uma outra forma de organização de termos, adequada para resolver esta questão. A estrutura emprega *modificadores* que são também mecanismos para indicar papeis dos termos incluídos na indexação.

5.3.1.3 Classes de indexação

Existem quatro tipos de indexação quanto à tradução:

- *Indexação por extração de informação*. Esta classe de indexação ocorre quando a *fonte* da análise é um documento de situações. Por exemplo, um catálogo de *bugs* em programas que rodam em uma empresa ou um cadastro de clientes bancários são fontes para análise e extração de termos representantes de situações.
- *Indexação por atribuição de termos*. Nesta classe de indexação, a fonte do conhecimento para a análise de situação são as próprias pessoas especialistas naquelas situações. Ela envolve a atribuição de *termos* (na criação dos casos) a partir de uma fonte outra, portanto, que não seja propriamente um documento textual.
- *Indexação automática* & *indexação manual*. Referem-se aos meios de processamento tanto da *indexação por extração* quanto da *indexação por atribuição*. Na indexação manual o especialista providencia ou a extração ou a atribuição de termos para caracterizar os atributos de casos. Na indexação automática, ao contrário, o próprio sistema de RBC tem a capacidade de, automaticamente, achar/determinar – a partir de um índice (ainda não organizado na forma de casos) mas previamente armazenado – aqueles *termos* a serem atribuídos/indexados aos casos que estejam em criação. A indexação usada em sistemas de RBC baseados em processos indutivos é um exemplo típico de indexação por atribuição automática (*feature selection*).

5.3.1.4 Indexação ≠ *Matching*

J. Kolodner – à parte a sua ambigüidade ao não rejeitar por completo a idéia de *índice* como ponteiro – também concebe a indexação como um processo incluindo três atividades básicas que seriam, nas suas próprias palavras [**KOL 96**, p. 355]:

- *Rotular* casos com seus conjuntos de características definidoras de situações (em nossa abordagem: *traduzir*, mediante *termos*, os atributos de situações); e
- *Atribuir valores de importância* para atributos/dimensões de uma representação;
- *Fazer matching* ou emparelhamento dos melhores casos (para efeito de *clustering* ou não).

Ora, as clarificações de conceitos de terceiros por Kolodner, além de não oferecerem alternativas conceituais, também não nos parecem bastante clarificadoras. A este respeito, estamos a afirmar que as definições de 5.1 a 5.5, por nós apresentadas, certamente vão se mostrar muito mais significativas para caracterizar todo o processo da indexação de casos do RBC do que apenas dizer que a indexação compreende as operações de rotular, atribuir valores e fazer *matching* por ela apontadas acima.

Além disso, ao incluir a operação de *matching* nos processos peculiares da indexação, uma tal inclusão certamente vai mais confundir o engenheiro de conhecimento do que clarificar. Note-se o seguinte: realizar *matching* ou emparelhamentos constitui um processo não somente do RBC mas se trata de um processo comum a diferentes áreas da computação onde, normalmente, o *matching* fica associado à operações de *resgate de informação* (cf. seção 2.7). Defendemos que esse processo de *matching* (como forma de resgate, propriamente) não faz parte da indexação de casos - como confusamente aparece em Kolodner [**KOL 96**, p. 355] – do mesmo modo que, em outras áreas da computação (em *Information Retrieval*, por exemplo) a indexação também não faz parte do resgate e vice-versa [**KOW 97**]. Em IA, a realização de *matching* constitui um processo próprio, inclusive com a sua denominação própria conhecida como *busca*, o que corresponde, particularmente no RBC, a *busca de casos* (em vez da indexação kolodneriana) – conforme as definições que seguem.

5.3.2 Outras clarificações

Também importantes são: *(i)* a busca funcionando como resgate de casos; *(ii)* a computação de similaridade, e *(iii)* o armazenamento de casos:

Definição 5.6: *Armazenamento de casos*. Armazenamento é o processo que consiste tanto na criação daquelas *estruturas de dados* para os casos quanto no uso destas estruturas de dados.

Definição 5.7: *Resgate de casos*. Resgate de casos, por outro lado, é um processo em três etapas fundamentais compreendidas pela *(i)* operação de indagação ou consulta; pela *(ii)* operação de busca de casos; e, por fim, pela *(iii)* operação de computação da similaridade.

Definição 5.8: *Indagação sobre casos*. A indagação a ocorrer no "pronto" de um sistema consiste *(i)* na ação de especificação de atributos conhecidos sobre casos; *(ii)* na ação de especificação de atributos adicionais sobre casos; e *(iii)* na ação de especificação de restrições para o resgate.

Definição 5.9: *Busca de casos*. A busca, propriamente, consiste nas funções de *matching* ou emparelhamento de casos e nos algoritmos de pesquisa.

Definição 5.10: *Computação de similaridade*. A computação da similaridade tem o sentido aqui adotado de mecanismo para estreitamento do espaço de soluções ou de busca de casos.

Note-se o sentido que a Definição 5.10 está emprestando à computação da similaridade. Em geral, a similaridade tanto pode ser computada para fazer retornar casos da memória – ou seja, como mecanismo propriamente de resgate (como visto na seção 2.7) – quanto pode ser usada para estreitar o espaço de uma solução (em tempo de *seleção de casos*). Por que então é necessário o estreitamento deste espaço de solução? Isto ocorre porque existe uma classe de resgate (por exemplo, a que usa *termos* de uma indexação) onde os casos são resgatados apenas com base em uma expectativa de similaridade, em virtude da sub-especificação da indagação, no ato do resgate. A sub-especificação na consulta resulta sempre na localização e no retorno de inúmeros casos. Nestas situações, a computação da similaridade tem o papel de fazer a descoberta dos melhores emparelhamentos entre estes inúmeros casos através do estreitamento do espaço de soluções.

5.4 Um tratamento da indexação baseado em tabela: modelo IBT

O quadro conceitual introduzido acima nos permite retomar aqui a questão (formulada na seção 5.1) sobre como operacionalizar a indexação de uma classe de casos que possam representar os objetos tverskyanos de nosso modelo de similaridade. *Indexação Baseada em Tabela* – ou abreviadamente *IBT* – constitui justamente esse modelo de indexação e de representação que propomos para oferecer resposta a esta questão. Ele foi concebido de modo a combinar três exigências básicas:

- A exigência de que casos do RBC funcionem como o mecanismo central de representação dos objetos tverskyanos;
- A exigência de que casos sejam indexados sob a forma de uma organização do tipo tabular, em apoio à computação de suas similaridades;
- A exigência de que casos – indexados via IBT– funcionem como algo parecido com uma estrutura de *frames*, como mecanismo para garantir a posterior computabilidade da similaridade proposta.

A seção 5.4.1 trata do surgimento da idéia de tabela para indexação de casos enquanto a seção 5.4.2 detalha esta forma de organização da indexação. A seção 5.5 trata do relacionamento entre IBT e o domínio selecionado para aplicação. A parte restante do capítulo (seção 5.6) explora formalismos alternativos de organização de casos.

5.4.1 Indexação tabular em Russell & Norvig

Indexar ou organizar em tabelas conhecimentos extraídos de situações é uma idéia intuitiva. Observe-se, por exemplo, a forma de organização dos termos na estrutura (4) da Figura 5.1. Esta organização compreende duas colunas: uma para os termos que indicam dimensões conceituais (*Sujeito* ou *Agente*, *Ação*, *Objetos*, *Modificador*), outra para valores destes mesmos termos.

Russell e Norvig, de um modo similar, discutem a implementação de bases de conhecimento –*não constituídas por casos* – também tomando por modelo uma organização tabular deste conhecimento [**RUS 95**, p. 301]. Ao tratarem dos *sistemas de raciocínio lógico*, concebem a indexação de sentenças lógicas fazendo uso de uma estrutura de dados conhecida como tabela *hash*, tendo em vista armazenar e também resgatar informação indexada por chaves fixas. O conteúdo das suas tabelas é formado, portanto, por sentenças lógicas do tipo sentenças literais *ground* (sentenças que não possuam variáveis).

Nossa indexação se fundamenta na idéia destes autores. Ela, porém, constitui um esforço para estender a indexação de sentenças lógicas desses autores para uma metodologia de indexação de casos baseada em tabela. Ou seja, ao contrário do modelo de Russell e Norvig para indexar *sentenças lógicas*, o modelo IBT trata de indexar *atributos de casos*. A questão central, no entanto, é a de como fazer essa indexação se ajustar ao conceito de objeto tverskyano, que constitui toda a base de nossa investigação.

Efetivar este ajustamento entre as nossas necessidades de estruturação de casos e a organização tabular constante em Russell e Norvig significa ter de alterar tanto a forma da organização tabular quanto os próprios conteúdos da informação indexada. O exemplo geral (sem conteúdo ou domínio particular) dessa indexação em tabela está mostrado na seção 5.4.2.3. Exemplos de indexação tabular particularizada para o domínio da análise de crédito são discutidos no Capítulo 7 dos experimentos.

5.4.2 IBT como representação de objetos tverskyanos

Como então indexar atributos de casos que levem em conta as propriedades dos objetos tverskyanos? Ou, dizendo de outro modo: como empregar esta indexação de casos para computar as suas

similaridades? Sabemos, pelo Capítulo 4, que a indexação ou criação de casos deverá operar de tal como a viabilizar a computação das similaridades; mas, como acoplar estruturalmente à indexação atributos de objetos, valores, diagnosticidade e pesos para valores?

5.4.2.1 Regra de ajustamento operacional ao modelo de Tversky

Em IBT, resolve-se este problema do acoplamento aplicando-se uma regra de ajustamento ao modelo teórico de Tversky. A regra básica que estamos a propor para guiar este processo de criação de casos através de uma indexação tabular é a seguinte: qualquer objeto tverskyano pode ser representado por um caso do RBC de estrutura tabular se, para cada propriedade observada neste objeto, também for criada uma correspondente representação desta propriedade na estrutura deste caso em criação. A implementação desta regra pode ser conseguida pelo procedimento dado a seguir.

5.4.2.2 Procedimento da indexação IBT

Casos da forma tabular são indexados ou criados pelo emprego dos seguintes quatro passos básicos:

Passo 1: *Identificação dos termos descritores de atributos.* Tanto métodos manuais quanto automáticos podem ser usados para selecionar-se ou atribuir-se termos a situações. Em muitos domínios de aplicação – por exemplo, cadastros, históricos diversos, e também catálogos de problemas ocorridos ao longo de operações – podem ser empregados pelos especialistas tendo em vista identificar termos descritores de atributos úteis à criação de casos.

Passo 2: *Determinação dos valores dos atributos.* Para cada atributo selecionado no Passo 1 reservado à eliciação (cf. seção 5.3.1.1), a ele deve ser atribuído o valor mais apropriado, quer seja ele um valor simbólico ou um valor numérico, ou booleano, ou ainda um valor textual.

Passo 3: *Atribuição de votos ou pesos aos valores de atributos.* Também, devem ser providenciados votos ou pesos para expressarem a importância de instâncias particulares de valores dos atributos em questão. Em nossas exemplificações e experimentos, pesos para expressarem a importância de valores devem ser atribuídos pelo especialista no domínio ou usuário do sistema. Métodos humanos de julgamento de importância podem estabelecer votos para valores de atributos em uma certa escala quantitativa de números de votos (uma escala de um 1 a 10, por exemplo). Esta escala de votos para valores de atributo, no entanto, vai depender das especificidades do domínio de aplicação do modelo IBT.

Passo 4: *Quantificação da diagnosticidade de atributo.* Diagnosticidade, como visto, mede a importância do atributo na definição da *natureza* dos objetos. Estamos também adotando que este tipo de julgamento deve ser mais apropriadamente feito pelo *designer* do sistema (e não mais pelo seu usuário), empregando para isto uma escala de atribuição de votos indicadores da essência dos objetos. Quanto maior a quantidade de votos atribuída ao nome de um certo atributo, maior será a diagnosticidade ou a essencialidade daquele atributo para a definição daquele objeto que está sendo representado mediante os casos.

5.4.2.3 Forma geral dos casos indexados por IBT

O procedimento descrito acima será capaz de dar origem a uma organização particular dos atributos de objetos e situações. A Figura 5.2 retrata esta forma geral de organização de atributos no interior dos casos a serem modelados. Os casos modelados na forma descrita nesta figura satisfazem àquela regra básica de ajustamento do modelo computacional ao modelo cognitivo de Tversky.

Caso j		
Diagnosticidade-atributo	Atributo : Valor	Votos-valor-atributo
D_1	$A_1 : V_1$	Vt_1
D_2	$A_2 : V_2$	Vt_2
...	...	...
D_n	$A_n : V_n$	Vt_n

Figura 5.2: Forma genérica dos casos baseados em tabela

Qualquer *caso j* a integrar a uma base de caso terá de exibir atributos de um espaço de atributos $<A_1, A_2, ..., A_n>$ caracterizadores do objeto *j*; valores de atributos $<V_1, V_2, ..., V_n>$; votos ou pesos $<Vt_1, Vt_2, ..., Vt_n>$ dados aos valores destes atributos; e, finalmente, a diagnosticidade $<D_1, D_2, ..., D_n>$ de cada atributo deste objeto *j*. Estamos estabelecendo que também esta diagnosticidade seja medida pelo projetista (e não pelo usuário do sistema) em termos de quantidade de votos.

A figura, portanto, ilustra a forma genérica dos casos a serem indexados via *IBT* e – como conseqüência desta organização – a forma dos casos apropriada aos modelos de similaridade *SIM(m,p)*, de *ranking ORDEN* já apresentados, e também do modelo *AVAL* de avaliação.

Tanto os aspectos da aplicação deste esquema de indexação e de organização de casos quanto os aspectos da comparação com outras metodologias computacionais são deixados para análise no Capítulo 7. Introduzimos, porém, na parte restante deste capítulo, o domínio de realização do modelo IBT e também elementos de comparação deste modelo com outros formalismos de organiza-

ção de casos.

5.5 Indexação IBT na computação de casos de *crédito*

Tversky e Gati desenvolveram os seus experimentos cognitivos tomando como centro das investigações dois grupos diferentes de objetos: *(i)* o grupo das *figuras geométricas* e *(ii)* o grupo de *países* de continentes diferentes. Ao contrário destes interesses, a realização computacional dos nossos modelos considera uma classe de objetos também real mas de muito maior complexidade, qual seja a classe dos *objetos financeiros*. Através desta realização computacional, procura-se enxergar a indexação e o RBC como estando a serviço da representação computacional da classe particular de objetos tverskyanos constituída pelos créditos ou empréstimos financeiros. Ou seja: tratar-se-á da aplicação da abordagem IBT de indexação, da similaridade e do *ranking* de casos ao domínio particular das decisões sobre *crédito* financeiro – um domínio anteriormente introduzido na seção 3.7.2.

Aplicar a indexação IBT a decisões sobre crédito ou empréstimo significa dizer que nosso interesse de indexação passa a se concentrar na criação de casos do RBC que devam exibir as seguintes propriedades:

- *Casos de crédito passam a ser objetos manipuláveis que devem encerrar alguma resposta em matéria de concessão de um crédito pretendido e de sua qualidade.* Como então encapsular no interior de casos aqueles componentes que venham a funcionar como *lições* (ou respostas) úteis sobre possíveis decisões de concessão ou não de créditos financeiros?
- *Casos de crédito passam a ser objetos cujas representações devem se enquadrar na forma geral IBT estabelecida para os objetos tverskyanos.* Como, conjuntamente, levar em conta *(i)* tanto as condições atreladas ou atributos associados a créditos passados, quanto *(ii)* os votos ou pesos associados a esses atributos e seus valores?
- *Casos de crédito passam a ser objetos cujas similaridades também devem ser computadas segundo o modelo SIM(m,p) estabelecido para os objetos tverskyanos, em geral.*

Casos de créditos modelados com estas propriedades permitem que estes objetos possam ser comparados tanto em relação *(i)* às similaridades das condições de operações de crédito ou similaridades dos atributos da concessão; quanto em relação *(ii)* às similaridades dos resultados das decisões tomadas/a tomar.

5.6 IBT e mecanismos alternativos de organização de casos

Para efeito de mostrar as diferenças entre o modelo IBT proposto e outros formalismos alternativos de indexação e organização de casos, a parte restante deste capítulo destaca 3 outras questões da

realização de casos: *(i)* a atribuição de pesos por métodos estatísticos, e não pelo especialista do domínio; *(ii)* a formalização em separado do vocabulário de casos; e *(iii)* a representação de casos em estruturas de *frame* (Quadros), recentemente redescritas por G. Bittencourt [**BIT 98**].

5.6.1 Atribuição de importância por método estatístico

Nos Passos 3 e 4 do procedimento descrito na seção 5.4.2.2, o julgamento sobre os pesos para valores e para a diagnosticidade de atributos é uma tarefa para o usuário e para o *designer* dos sistemas, respectivamente. Deixa-se em aberto, porém, a possibilidade de automação deste processo através, por exemplo, de uma abordagem estatística. Nesta hipótese, metodologias estatísticas já disponíveis para a visualização de conteúdos de bancos de dados textuais (*Conceptual Space Visualizer*) podem ser reutilizadas para atribuir pesos ($w_{i,j}$) ao *valor j de um atributo* do *objeto i* entre *n* objetos ou casos através dos seguintes passos [**SUG 98**]:

1: Cálculo das densidades d_{ij} e $\overline{d}_j$ do *valor j* de um atributo no objeto *i* e também em *n* objetos, respectivamente:

$$d_{i,j} = \frac{\textit{Freqüência do valor } j \textit{ do atributo no objeto } i}{\textit{Freqüência de todos os valores de atributo no objeto } i}$$

$$\overline{d}_j = \frac{\textit{Freqüência do valor } j \textit{ do atributo em } n \textit{ objetos}}{\textit{Freqüência de todos os valores de atributo em } n \textit{ objetos}}$$

2: Cálculo da distribuição do *valor j* (ou v_j) de um atributo:

$$v_j = \frac{\sum_{i=1}^{n} (d_{i,j} - \overline{d}_j)^2}{n-1}$$

3: Cálculo, propriamente, do *peso para o valor j* (ou w_{ij})de um atributo do *objeto i*:

$$w_{ij} = \frac{d_{i,j} \times v_j}{\overline{d}_j^{\,2}}$$

A atribuição de pesos, estatisticamente, permite automatizar esta parte da indexação e traz vantagens sobre a indexação por especialistas, tais como um menor custo de indexação, menor tempo de processamento e até mesmo uma maior consistência entre os termos selecionados para indexação e os objetos sendo indexados. No entanto, a indexação por especialista acarreta uma maior vantagem em relação a uma maior habilidade em determinar abstrações conceituais e em julgar o valor dos termos/conceitos a serem incluídos como componentes dos casos.

5.6.2 Vocabulário como ontologia: A *Lógica das Descrições*

Kolodner indaga a si própria sobre a questão "*de onde vem o vocabulário para indexação*" (*Where does Vocabulary for Indexing Come From*?) [**KOL 96**, p. 357], uma questão por nós já respondida na seção 5.3.1.1. Interessa-nos, neste ponto, a indagação contrária ao problema de Kolodner: *Para onde deve ir o vacabulário atribuído/extraído de situações?* Propõe-se que este vocabulário possa convergir para uma estrutura – uma espécie de *ontologia* – a ser representada em uma linguagem formal, orientada ou especializada em lidar com terminologias, tal como a *Lógica das Descrições* [**BAA 99, RUS 95**, p. 323]. A vantagem principal em se formalizar terminologias de situações está em evitar que o sistema venha a cometer ambigüidades no emprego dos termos próprios de um domínio particular e, sobretudo, possui a vantagem de proporcionar a esse sistema o compartilhamento de conceitos comuns sobre objetos representados como casos.

5.6.3 Vocabulário como ontologia: exemplo em *help desk*

Suponha que o domínio de aplicação do *Web Help Desk* da seção 3.7.1seja o apoio em depuração de software, para continuar com nossas exemplificações da seção 5.3.1.2. O sistema terá de conhecer as muitas classes de erros que podem infestar um programa imperativo. Possíveis ações de um sistema *Web Help Desk* vão depender do seu conhecimento sobre o significado destas terminologias envolvendo estas classes de erro. Como então formalizar essas terminologias de erros a integrarem os casos de apoio da arquitetura de *help desk*? Como representar, em quadros, casos de erros de programação?

Orientada para esses sistemas de assistência ao usuário, a Figura 5.3 ilustra uma ontologia de erros de programação imperativa representada em Lógica das Descrições (e onde os símbolos próprios para os operadores conectivos desta lógica foram substituídos pelos símbolos $\wedge$ e $\vee$).

A descrição constante nessa figura ilustra uma tipologia de erros, sendo as suas duas classes principais, os erros internos (por violação da linguagem de programação, propriamente) e os erros externos ao programa (como os devidos a pane de *hardware*, documentação errada, aproximações indevidas por parte do algoritmo, etc.). Erros de *longa* e de *curta distância*, por outro lado, são erros léxicos ou sintáticos tomados em relação ao local do aparecimento dos seus efeitos em um programa; esses efeitos podem surgir mais distantes ou menos distantes da fonte propagadora dos erros.

Uma representação de vocabulário tal como esta pode ser relevante para a própria representação de casos se os mecanismos de inferência do sistema providenciarem a visitação tanto à representação do vocabulário quanto à representação dos casos.

erro-qualquer := *programa-imperativo* ∧
∃ *erro*. (*erro-externo* ∨ *erro-interno*)

algum-erro-externo := *programa-imperativo* ∧
∃ *erro*. *erro-externo*

algum-erro-interno := *programa-imperativo* ∧
∃ *erro*. *erro-interno*

erro-externo := *programa-imperativo* ∧
∃ *erro*. (*hardware* ∨ *documental* ∨ *conceitual* ∨ *teleológico* ∨ *aproximativo*)

erro-interno := *programa-imperativo* ∧
∃ *erro*. (*léxico* ∨ *sintático* ∨ *semântico*)

algum-erro-semântico := *programa-imperativo* ∧
∃ *erro*. (*estrutural* ∨ *característico* ∨ *violação-memória*)

algum-erro- léxico := *programa-imperativo* ∧
∃ *erro*. (*longa-distância* ∨ *curta distância*)

algum-erro-sintático := *programa-imperativo* ∧
∃ *erro*. (*longa-distância* ∨ *curta-distância*)

Figura 5.3: Terminologia sobre erros de programação em *Lógica das Descrições*

5.6.4 Vocabulário, quadros e casos de *help desk*

Selecione-se, de início, um daqueles erros de programação constante na terminologia anterior, por exemplo, *Erro-característico* para sua representação em casos na forma de quadros. Quadros são recursos computacionais para modelarem objetos que possuem identidade própria – em um certo domínio – tais como erros de programação. A modelagem dos erros na forma de casos representados em quadros permitirá ao usuário fazer indagações, permitirá as modificações dos conhecimentos encapsulados e as inferências. Os quadros são construídos utilizando-se três elementos básicos: *(i) atributos*; *(ii) relações*; e *(iii) procedimentos*. Os atributos a comporem os *quadros* correspondem às propriedades dos objetos (erros, no exemplo) que são relevantes para uma dada aplicação, tal como em *help desk* para apoio em software. *Termos* vão descrever o nome das propriedades ou atributos associados aos objetos. As *relações* determinam as ligações semânticas entre os diferentes objetos do domínio. Os relacionamentos descritos são especialmente *generalização* e *associação*. Os *procedimentos* cumprem o papel de manipulação dos objetos representados, ou seja, toda e qualquer operação sobre os objetos (consulta, modificação, exclusão).

Erro-característico, por sua vez, é um tipo de erro semântico em linguagens imperativas (constante no vocabulário da Figura 5.3), que pode ocorrer, por exemplo, em instruções *case* (da forma: ***case*** *expressão-ordinal* ***of*** *lista-do-case*; ... [;] ***end***, onde *lista-do-case* ::= *lista-constantes* : *instrução*). Trata-se de erros mais induzidos pelas características próprias de alguma linguagem de programação (de onde vem o nome do *bug*) do que pela imperícia do programador.

Quadros representando casos de erros de programação estão mostrados na Figura 5.4. Por definição, em sistemas de *help desk* dessa natureza, a meta dos casos representados é poder oferecer algum tipo de resposta que funcione como assistência à correção de programas, como no projeto SQUAD japonês, com 20.000 casos de *bugs* (cf. seção 1.2). Na Figura 5.4, um erro do tipo característico está representado no quadro *erro-característico* com seus atributos particulares, que são: *(i)* a fonte do problema está na instrução *case*; *(ii)* o tipo do problema; *(iii)* característica teórica peculiar; *(iv)* sugestão de ajuda para a correção deste problema. O quadro *erro-característico*, além desses atributos, vai herdar os atributos do quadro *algum-erro-semântico*, que são: *(i)* ocorrência do erro em tempo de execução; *(ii)* sem mensagem de erro ou com mensagem cifrada; *(iii)* resultando em parada súbita da execução do programa.

A assistência providenciada pelos casos do exemplo se refere ao problema do emprego da instrução *case* quando não tiver sido prevista qualquer correspondência entre a *expressão-ordinal* da sintaxe do *case* com quaisquer dos seus rótulos ou *lista-do-case*. Para este erro, o suporte ao programador vem, na forma de texto, como sendo uma faceta do atributo *Sugestão de Ajuda.* As facetas, por conseguinte, especificam os tipos de valores esperados de atributo e, se forem necessários, os procedimentos adequados para calcular o valor de um certo atributo. Como pode ser visto, trata-se de uma representação de casos bem diferenciada do modelo IBT.

Quadro: *algum-erro-semântico*	***Super-quadro***: *erro-qualquer*		
ATRIBUTOS	DEFAULT	TIPO	SE-NECESSÁRIO
Tempo ocorrência	Execução	Símbolo	
Mensagem de erro	Cifrada	Símbolo	
Resultado	Parada súbita	Símbolo	

é_um

Quadro: *erro-característico*	***Super-quadro***: *algum-erro-semântico*	
ATRIBUTOS	DEFAULT	TIPO
Fonte do problema	Instrução *case*	Símbolo
Tipo do problema	Não lógico	Símbolo
Sugestão de ajuda	Verificar se expressão do *case* corresponde aos seus rótulos	Texto

Figura 5.4: *Quadros* representando um caso de apoio em programação

5.7 Conclusão

Este capítulo se concentrou em duas questões fundamentais do raciocínio baseado em casos: *(i)* a questão de como providenciar uma extensão computacional para um modelo cognitivo já existente, através de uma representação de casos para os objetos tverskyanos; essa primeira questão nos remete então para um outro problema dela decorrente; *(ii)*a questão fundamental da indexação em RBC. O capítulo parte da análise de conceitos correntes e concepções errôneas sobre indexação envolvendo *índice*, *matching*, *indexação*, e propõe então novas interpretações com base nos processos verificados no domínio da área de *Information Retrieval*. Em seguida, o capítulo se concentra na proposta de um modelo de organização de atributos e de criação de casos denominado de *Indexação Baseada em Tabela* (IBT). O modelo IBT tem inspiração em material encontrado em Russell e Norvig mas se presta a finalidades diferentes. Em vez da indexação de *sentenças lógicas* procura-se organizar, em forma tabular, toda aquela informação necessária à computação subseqüente das similaridades de casos criados com a IBT. Aspectos de aplicação são deixados para o Capítulo 7, após já ter sido indicado, neste capítulo (seção 5.5), aquilo a ser perseguido com os experimentos de indexação e de similaridade de casos na área das decisões sobre créditos financeiros.

Capítulo 6

Avaliação qualitativa de casos baseada em métricas

6.1 Introdução

6.1.1 Motivações e razões para avaliar casos

Avaliar sistemas modelados com tecnologia de RBC constitui um daqueles problemas ainda em aberto no contexto do RBC onde, praticamente, inexistem contribuições de tipo mais operacionais. A este respeito, os próprios trabalhos de Janet Kolodner são pouco expressivos [**KOL 93**]. Em seu texto básico, algumas diretrizes podem ser encontradas, de modo um tanto disperso, sobre o problema. A autora, no Capítulo 15 sobre a "*construção de um raciocinador baseado em casos*", do referido texto tão somente estabelece considerações sobre a avaliação destes sistemas de RBC (considerações 1 e 2). Ela assevera que a alta complexidade de se avaliar as soluções encontradas por um racionador baseado em casos tem como causa os vários tipos diferentes de conhecimento que se fazem necessários para a tarefa [**KOL 93**, p. 542]:

> "É muito difícil definir precisamente cada um destes tipos de conhecimentos"

Isto talvez venha a explicar a carência de trabalhos científicos significativos nesta área específica da recente tecnologia de RBC, o que contrasta com avançadas metodologias já existentes para avaliações em áreas como a de Banco de Dados [**FIR 95, MOT 97**], de sistemas *Query Answering* [**GOD 99, AND 97**, p.2] e de *Information Retrieval* [**KOW 97, SAL 75**]. Não obstante a complexidade da tarefa, avaliar aplicações de RBC – E NÃO APENAS AS SUAS FERRAMENTAS, como no Relatório da *AI Perspectives* [**ALT 95**] – se torna uma exigência dos processos de RBC por diferentes razões, como as que enumeramos na seqüência:

- Pode-se querer determinar a *qualidade* de ambientes baseados em casos já construídos, para que se possa conferir o grau de satisfação dos vários objetivos de uma base de casos;
- Pode-se querer introduzir inovações em ambientes já existentes e a avaliação vai determinar esta viabilidade de inovações;

- A avaliação também se torna necessária sempre que se tem de comparar a performance de diferentes *designs* baseados em casos;
- Finalmente, requer-se ainda uma avaliação sempre que as características ainda não conhecidas de um ambiente de casos devam ser comparadas com características já bem estudadas em outros ambientes.

O presente capítulo trata de um *framework* para o enfrentamento desta tarefa. O *insight* para uma abordagem desta natureza poderia provir quer do tratamento da questão em Banco de Dados, quer de *Query Answering* ou ainda da área de *Information Retrieval* (IR). A área de *Information Retrieval*, porém, parece apresentar maior solidez e uma maior formalização das suas metodologias avaliativas. A nossa opção recai sobre este último paradigma como a fonte de motivações para a abordagem, aqui discutida.

6.1.2 Objetivos da abordagem *AVAL*

Em nossa abordagem, avaliar *qualitativamente* modelos baseados em casos significa explorar, de uma forma ou de outra, três questões básicas:

- Até que ponto um certo modelo baseado na tecnologia de RBC atende aos seus objetivos estabelecidos em relação ao usuário?
- Quais as principais razões para o possível insucesso em resgatar casos *relevantes* para o usuário?
- Quais as razões para o insucesso em rejeitar casos *não relevantes*?

São estas as questões básicas envolvendo o desempenho dos sistemas construídos com tecnologia de RBC e cujo tratamento estamos a denominar de concepção *AVAL*. Fundamentalmente, *AVAL* constitui uma proposição de avaliação de bases de casos centrada em *métricas* com a meta de favorecer *insights* sobre o desempenho qualitativo destes sistemas. Uma tal visão de avaliação orientada para *insight* vai contrastar, por exemplo, com possíveis abordagens mais voltadas para operações de *testes exaustivos* de sistemas.

Nas seções 6.2.1 e 6.2.2 seguintes, procura-se caracterizar, de um modo geral, aqueles mecanismos internos dos modelos de RBC que estão na origem da necessidade de avaliação, quais sejam: *(i) indagações, (ii) espaço de respostas, (iii) respostas resgatadas* × *não resgatadas, (iv) relevantes* × *não relevantes*. São discutidos, ainda, conceitos basilares que estamos a identificar para fundamentar a visão de avaliação ora proposta. Entre estes conceitos identificados como indispensáveis estão os de: *(i) efetividade & eficiência; (ii) avaliação orientada para provas* × *avaliação orienta-*

da para insight; *(iii) macro avaliação* × *micro avaliação*; e *(iv) critério quantitativo*. Finalmente, as seções 6.3 a 6.8, mais diretamente, se concentram na especificação daquelas propriedades da avaliação de casos constitutivas da visão *AVAL*, com destaque para a definição dos critérios avaliativos e para a definição das métricas fundamentais necessárias para esta tarefa.

6.2 Conceitos basilares no modelo *AVAL*

6.2.1 Espaço de respostas

Modelos baseados em casos têm como meta acessar um espaço de casos-resposta e fazer retornar um conjunto de casos a satisfazerem uma certa indagação do usuário. Processos de avaliação têm como objetivo justamente medir este grau de satisfação do usuário (quando esta avaliação não esteja, especificamente, orientada para a gestão do sistema). A Figura 6.1 ilustra este espaço de respostas particionado, neste exemplo particular, em 14 casos que estão a representar uma base de casos em um domínio de interesse de uma população de usuários.

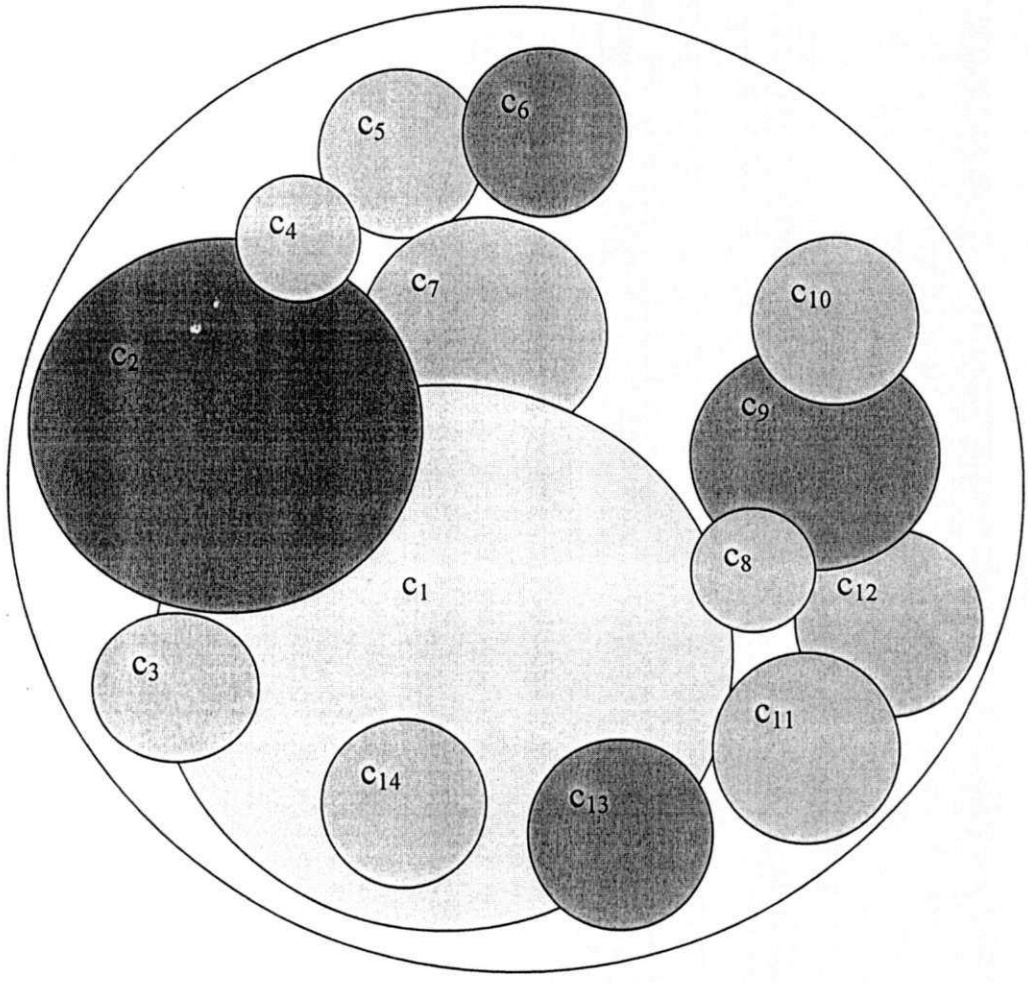

Figura 6.1: Base de casos como espaço de respostas

Quando o usuário decide formular uma indagação em busca de casos similares ao que tem em mãos, uma base de casos total (como a da Figura 6.1) fica logicamente dividida em quatro diferentes segmentos conforme mostrado na Figura 6.2.

Figura 6.2: Efeito de indagações sobre o espaço de respostas

Casos relevantes, por definição, são aqueles que contém informação capaz de ajudar ao indagador a responder a sua questão. *Casos não relevantes*, ao contrário, são aqueles que não propiciam, nem direta nem indiretamente, qualquer informação útil a uma indagação. Existem duas possibilidades em relação a casos relevantes e não relevantes. Eles são passíveis de se tornar *casos resgatados* ou *casos não resgatados* através de indagações.

São estes fenômenos do RBC mostrados na Figura 6.2 que estão na base dos processos de avaliação da qualidade de casos e cuja análise requer um amplo aparato de conceitos de avaliação que variam de conceitos cognitivos, a conceitos econômico-financeiros, passando pelos conceitos tecnológicos, propriamente.

6.2.2 Avaliação de respostas: definições

Destacamos, prioritariamente, os conceitos analisados na seqüência **[SAL 75]**.

Definição 6.1: *Efetividade* ("*Effectiveness*") é a capacidade de uma base de casos em atender aos propósitos para os quais ela tenha sido desenhada. Isto é, efetividade tem a ver com a capacidade de fornecer à população de usuários aqueles casos-resposta de que esta população necessita.

Definição 6.2: *Eficiência* ("*Efficiency*"), por sua vez, é uma medida em termos de *custo* e, às vezes, em termos de *tempo* necessários para a realização de um conjunto de tarefas de uma base. Um completo processo de avaliação leva em conta tanto a eficiência quanto a efetividade das bases de casos.

Definição 6.3: *Avaliação Orientada para Insight* se contrapõe à *Avaliação Orientada para Prova*. Nesta última classe de avaliação, os resultados da avaliação são produzidos para servirem de testes ou para serem submetidos a provas em sistemas da vida real. A primeira classe de avaliação, por

outro lado, dispensa, como o nome denota, que os seus resultados sejam submetidos a provas ou comprovações empíricas. Avaliações orientadas para *insights* podem inclusive utilizar-se de algum *recurso de simulação* de baixo custo.

Definição 6.4: *Micro-Avaliação*, por sua vez, se contrapõe a *Macro-Avaliação*. Nesta última classe de avaliação, a ênfase está: *(i)* na análise das indagações (*entradas*); *(ii)* na questão do tempo; *(iii)* e na análise das respostas obtidas (*saídas*). O processamento, propriamente, é então visto como uma caixa-preta, na avaliação do tipo *macro*. Na micro-avaliação, porém, a ênfase está na análise de algum componente do sistema, em particular. Por exemplo, os mecanismos de indexação, e da similaridade de casos, separadamente, podem ser objetos de micro-avaliações.

Definição 6.5: *Critérios Quantitativos* são requisitos para obtenção de medidas da performance de bases de casos. A definição destes critérios quantitativos pode ser feita com fundamentos em: *(i)* critérios já existentes para sistemas já em operação; *(ii)* propriedades dos processos componentes dos sistemas de RBC; *(iii)* técnicas de gestão existentes para análise de sistemas. Tanto os critérios definidos, quanto as interrelações entre parâmetros vão contribuir para a construção de *métricas* que tenham como objetivo a *quantificação da relevância* dos casos de um sistema de RBC.

A proposição *AVAL* defende o uso de critérios quantitativos (métricas) para avaliar a qualidade de casos do RBC com base em uma crença: a crença de que subjacentes à esta proposição estão as relações entre *quantidade* e *qualidade*. A quantificação da relevância tem o papel de expressar a qualidade de uma base de casos, ao aplicarmos em *AVAL* a relação estabelecida por Abraham Kaplan (In: [**SLI 97**, p. 19]):

> *"As quantidades são quantidades de qualidades e a qualidade medida tem apenas a grandeza expressa em sua medida (...). A tranformação de quantidade em qualidade ou vice-versa é um processo lógico ou semântico, e não uma questão de ontologia"*.

6.3 Parâmetros de avaliação em *AVAL*

Vamos considerar, nesta seção, os critérios de avaliação quando esta avaliação interessa, sobretudo, a uma população de usuários de uma base de casos (Outras avaliações/outros critérios – tais como critérios econômicos – podem servir prioritariamente, por exemplo, aos gestores de uma base de casos). Entre os vários possíveis critérios deste tipo, seis (06) deles podem ser considerados como sendo críticos para efeito de avaliação de casos:

(i) *Proporção retorno*. Isto é, a proporção que mede a capacidade que um sistema possui de exibir todos os casos relevantes para uma dada consulta empreendida.

(ii) *Proporção de precisão*. Isto é, a proporção que mede a capacidade que um sistema possui de desprezar todos os casos não relevantes para uma certa indagação.

(iii) *Esforço do usuário*. Significa o esforço intelectual ou físico, requerido dos usuários *(i)* ao formularem suas indagações, *(ii)* ao instrumentarem o próprio processo de busca, ou mesmo *(iii)* ao examinarem o *output* dos resgates.

(iv) *Tempo de resposta*. Significa o espaço decorrido entre a aceitação de uma indagação do usuário pelo sistema e o *display*, propriamente, das respostas do sistema.

(v) *Apresentação*. É a forma de visualização do *output* dos resgates e que também pode influenciar a capacidade do usuário em fazer uso daqueles casos que forem resgatados.

(vi) *Cobertura da coleção de casos*. Isto é, até que ponto todos os casos potencialmente relevantes para o usuário venham a estar devidamente incluídos numa base de casos, em um certo domínio.

De entre estes seis critérios de avaliação, três deles podem ser identificados como sendo os de mais fácil *mensuração*, relativamente. São eles:

- *Esforço do usuário* pode ser expresso, por exemplo, quer em termos do tempo necessário para a formulação das indagações, quer em termos do tempo para uma interação com o sistema, quer ainda em termos do tempo para o exame de seu *output*;
- *Tempo de resposta*, por sua vez, é também um parâmetro diretamente mensurável. É o tempo tomado para executar buscas de casos. Existe porém uma ambigüidade na definição do que seja o tempo final de busca. É fácil definir o tempo de início de uma busca mas o tempo de término da busca é sempre afetado pelos pontos de vista particulares dos usuários de casos.
- *Cobertura da coleção de casos* também pode apresentar dificuldades de avaliação em termos de métrica. Isto acontece quando o domínio de aplicação for um domínio de verdade, em oposição a um domínio de brinquedo (*toy domains*), o qual não tenha sido, *a priori*, identificado como sendo altamente intensivo ou rico de conhecimentos. Nesta hipótese, os mecanismos disponíveis de eliciação de casos devem levar a uma estimação da abrangência desta coleção de casos que está sendo construída.

Excluídos estes três parâmetros por oferecerem menores problemas, somente os critérios da *proporção de retorno* e da *proporção de precisão* ("*recall*" e "*precision*", como respectivamente denominados na literatura de *IR*) devem apresentar maiores dificuldades de mensuração, como vem demonstrando a prática de *IR*. Tratamos particularmente destas mensurações de performance, na

visão do RBC.

6.4 Métrica de precisão e métrica de retorno

Duas condições iniciais são necessárias para que as proporções de *retorno* e de *precisão* de casos em uma base de casos possam ser quantificadas:

- *Contabilização*, separada, do conjunto dos casos resgatados e do conjunto dos não resgatados;
- *Existência de procedimentos* para se separar os casos relevantes daqueles casos não relevantes para certas indagações (a despeito da subjetividade da tarefa).

Satisfeitas estas condições, define-se então [**KOW 97**]:

$$Retorno = \frac{Quantidade\ de\ resgatados\ relevantes}{Quantidade\ de\ possíveis\ relevantes}$$

$$Precisão = \frac{Quantidade\ de\ resgatados\ relevantes}{Quantidade\ total\ de\ resgatados}$$

sendo que *Quantidade de possíveis relevantes* designa o número total de casos relevantes previstos na base de casos; *Quantidade total de resgatados* designa o número total de casos resgatados num arquivo de resultados de buscas e, finalmente, *Quantidade de resgatados relevantes* designa o número de casos resgatados que são realmente relevantes para as necessidades de um usuário.

Retorno, por definição, é uma medida da capacidade de encontrar todos os casos relevantes que estejam na base de casos; mede, portanto, a capacidade do sistema de resgatar casos relevantes. Enquanto que, *precisão* é uma medida da *acuidade* ou acurácia do processo de busca; mede a capacidade do sistema de rejeitar casos não relevantes. Necessidades de casos-resposta podem variar de usuário para usuário. Alguns usuários podem requerer uma alta probabilidade de retorno, isto é, o resgate de quase tudo que presumivelmente possa ser de seu interesse; enquanto outros usuários podem preferir uma alta probabilidade de precisão, isto é, a rejeição de tudo que possivelmente possa ser inútil. "Tudo mais permanecendo constante" (condição *coeteris paribus*), um bom sistema de RBC será aquele capaz de exibir tanto uma alta probabilidade de retorno quanto uma alta probabilidade de precisão.

6.4.1 Funcionamento das métricas

Suponha que os dois casos C_1 e C_2 tenham sido resgatados em resposta a um conjunto de indagações. Especificamente, suponha que o caso C_1 tenha sido o retorno para um total de 150 indagações

(150 emparelhamentos) e que o mesmo caso também tenha sido considerado pelo usuário como um caso relevante para 90 destas indagações. Suponha que o caso C_2 tenha sido retornado para 200 indagações e seja relevante para 100 delas. A Figura 6.3 ilustra esta aplicação das métricas em apreço.

Nome dos casos	Indagações emparelhadas	Quantidade relevantes	Quantidade irrelevantes
Caso C_1	150	90	60
Caso C_2	200	100	100

Figura 6.3: *Feedback* do usuário sobre resultados de indagações

Do ponto de vista da métrica de retorno, o caso C_2 será preferível ao caso C_1 uma vez que, em relação ao caso C_2, mais usuários o consideraram relevante ou mais usuários receberam o que dele pretendiam (100 versus 90). Contrariamente, a precisão de 60% do caso C_1 (*Precisão = Resgatados Relevantes/Quantidade de emparelhamentos* = 90/150 = 60%) será melhor do que a precisão de 50% do caso C_2.

6.4.2 Métricas de precisão média e retorno médio

Métricas como as apresentadas no início desta seção trabalham com valores *totais* obtidos após um número *k* de indagações (um conjunto dado de indagações) e uma coleção dada de casos. É possível, no entanto, trabalhar com o valor médio de precisão e o valor médio de retorno. *Precisão média* e *retorno médio* têm a vantagem de refletir aquela performance que o *usuário médio* pode esperar obter de um certo sistema de RBC. Primeiro, vamos definir os seguintes parâmetros:

a_i – como sendo a quantidade de casos resgatados e relevantes para a indagação *i*.

b_i – como sendo a quantidade de resgatados mas não relevantes para a indagação *i*.

c_i – como sendo a quantidade de relevantes mas não resgatados para a indagação *i*.

Retorno R_i e Precisão P_i correspondentes a cada indagação *i* individual, podem ser definidos como:

$$R_i = \frac{a_i}{a_i + c_i}$$

$$P_i = \frac{a_i}{a_i + b_i}$$

Medidas relativas ao usuário médio podem ser calculadas tomando-se a média aritmética de P_i e R_i do seguinte modo:

$$R_{\text{médio}} = \frac{1}{k}\sum_{i=1}^{k}\frac{a_i}{a_i + c_i}$$

$$P_{\text{média}} = \frac{1}{k}\sum_{i=1}^{k}\frac{a_i}{a_i + b_i}$$

6.4.3 Problemas com precisão e retorno

Precisão e *retorno* são, portanto, métricas que formam uma base inicial para se avaliar a *efetividade* de bases de casos. A Figura 6.4 apresenta as propriedades básicas da *precisão* (linha contínua) e do *retorno* (linha descontínua), em situações ideais. No eixo horizontal estão os N casos relevantes, resgatáveis em uma base de casos.

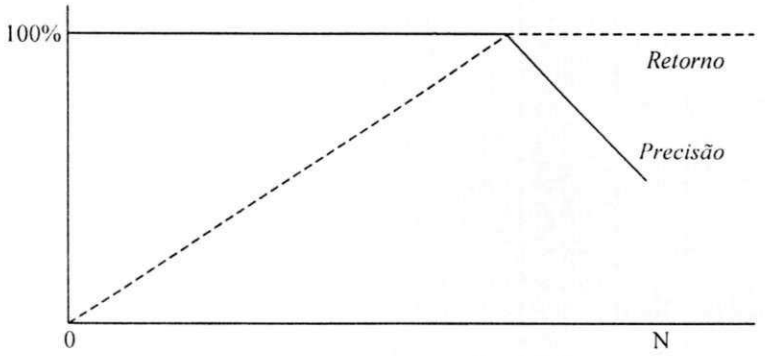

Figura 6.4: Precisão e retorno ideais

Precisão começa em 100% e mantém-se assim enquanto casos relevantes sejam resgatados. *Retorno* começa com valores próximos a zero e aumenta enquanto casos relevantes sejam resgatados e até que todos esses casos relevantes sejam esgotados. Uma vez os N casos relevantes tendo sido resgatados, os únicos casos a continuarem sendo resgatados serão aqueles não relevantes. *Precisão* então é diretamente afetada pelo resgate de casos não relevantes e cai para valores próximos de zero. A métrica de *retorno*, ao contrário, não é afetada pelo resgate de não relevantes e, uma vez havendo atingido a marca de 100%, ela aí permanece.

Essas métricas, contudo, apresentam ambigüidades matemáticas e também carecem de um certo paralelismo entre suas propriedades. Em ambientes controlados, será possível obter-se valores para ambas as métricas e relacionar uma com a outra. Nas respectivas fórmulas, valores como *quantidade de resgatados relevantes* e *quantidade total de resgatado* estarão sempre disponíveis. Porém, valores para *quantidade de possíveis relevantes* serão de mais difícil obtenção, em ambientes não controlados. Isto porque este componente das fórmulas sugere que todos os casos de uma base de casos já sejam conhecidos por quem faz a avaliação. Sugestões sobre como superar esta dificuldade

podem ser encontradas em Gerald Kowalski, aplicadas, evidentemente, ao contexto dos sistemas de *Information Retrieval* [**KOW 97**]. Um problema, porém, ainda persistirá: quais os valores das métricas de *retorno* e de *presisão* quando nenhum caso relevante for encontrado entre aqueles resgatados (na primeira situação) ou ainda quando nenhum caso for resgatado da base de casos (na segunda situação)? Em ambas as situações, as fórmulas matemáticas destas métricas tornam-se 0/0.

6.5 Métrica de refugo

A métrica de *refugo* é uma outra medida também interessante para a avaliação de casos porque ela mede a eficiência de um sistema relacionando-se diretamente com os casos *não relevantes* que venham a ser resgatados. Sua definição foi oferecida por G. Salton (In: [**KOW 97**, p. 230]):

$$Refugo = \frac{Quantidade\ de\ resgatados\ não\ relevantes}{Quantidade\ total\ de\ não\ relevantes}$$

onde *quantidade total de não relevantes* designa o total de casos não relevantes presentes numa base de casos. A métrica de *refugo* pode ser vista como a probabilidade de que um caso resgatado seja não relevante. Ela é o inverso da métrica de *Retorno* (a probabilidade de que casos resgatados sejam relevantes) e nela nunca vai-se ter a situação de 0/0, a não ser que todos os casos presentes em uma base de casos venham a ser relevantes para uma certa indagação. Um sistema ideal, portanto, será aquele que demonstre ter o máximo de *retorno* e o mínimo de *refugo*. A combinação de ambas as métricas, implicitamente, implicará em *precisão* máxima.

De entre as três métricas, *retorno*, *precisão* e *refugo*, esta última é a menos sensível à acurácia do processo de busca. Um valor maior para o denominador de *refugo* vai requerer mudanças significativas na quantidade de casos resgatados para poder afetar um certo valor corrente da métrica.

6.6 Métrica de retorno de relevância única

Retorno de relevância única ("Unique Relevance Recall") é uma medida interessante porque ela permite *comparar* dois ou mais algoritmos ou sistemas de RBC. Suponha um sistema implementado segundo algum método de indexação de casos ou segundo algum algoritmo de similaridade, ou ainda segundo alguma especialização do RBC, diferentemente de um outro sistema que implementa um método alternativo de indexação, por exemplo. A métrica de *retorno de relevância única* permite então medir a quantidade de casos relevantes que sejam resgatados por um certo algoritmo mas que não sejam resgatados por outros algoritmos. Ela é definida como:

$$Retorno\ de\ Relevância\ Única = \frac{Quantidade\ de\ relevantes\ únicos}{Quantidade\ de\ relevantes}$$

Quantidade de relevantes únicos designa a quantidade de casos relevantes que não foram resgatados por outros algoritmos. Por sua vez, o denominador *quantidade de relevantes* pode assumir duas interpretações, a depender do objetivo da avaliação. Esse denominador pode significar *(i)* ou a quantidade total de casos relevantes resgatados por todos os algoritmos (*quantidade total de resgatados relevantes*); *(ii)* ou pode significar a quantidade total dos casos individualmente resgatados por cada algoritmo (*quantidade total de resgatados relevantes únicos*). Na hipótese de se empregar o denominador *(i)*, a métrica *retorno de relevância única* vai medir, de um total de casos resgatados, o percentual daqueles que foram resgatados em função de cada algoritmo ou sistema, contando-se, inclusive, os casos resgatados em comuns. Na hipótese de se empregar o denominador *(ii)* a mesma métrica vai medir, sobre o total dos resgatados, o percentual daqueles casos relevantes, resgatados em função de cada algoritmo especificamente, mas sem incluir os resgates comuns aos algoritmos.

6.7 Métrica de utilidade

Alguns sistemas têm o papel de alertar o usuário sempre que eles venham a encontrar informação de potencial interesse para o usuário. Entre eles encontram-se sistemas tais como: *(i)* os agentes inteligentes, *(ii)* os filtradores de texto, e *(iii)* os categorizadores de dados. Esses sistemas agem sem qualquer *input* humano e suas decisões são binárias, por narureza: ou eles alertam o usuário ou eles ignoram a informação que eles venham a encontrar. A métrica denominada *métrica de utilidade* foi criada com o papel de avaliar a efetividade destes sistemas. Apropriada ao RBC, ela avalia a capacidade de encontrar e de alertar ou não sobre possível informação de utilidade. Ela foi proposta por William Cooper ainda em 1973 ("*On Selecting a Measure of Retrieval Effectiveness*") e está definida em **[KOW 97]**:

$$\text{Utilidade} = \alpha*(\textit{Relevantes Resgatados}) + \beta*(\textit{Não Relevantes Não Resgatados}) - \delta*(\textit{Não Relevantes Resgatados}) - \gamma*(\textit{Relevantes Não Resgatados})$$

Aqui, α e β são pesos positivos com os quais o usuário marca aqueles casos relevantes resgatadas e aqueles casos não relevantes mas que não foram resgatadas. Enquanto isto, δ e γ são pesos negativos para marcar aquelas informações não relevantes mas resgatadas como ainda aquelas informações relevantes mas que não foram resgatadas.

6.8 Métrica de eficiência

A avaliação da *eficiência* dos sistemas de acesso à informação, em geral, é muito mais difícil e se encontra menos avançada do que a avaliação da *efetividade*, mostrada acima. Isto porque a análise da eficiência demanda dados acurados sobre os *custos* dos sistemas, sobre os seus *benefícios* e ain-

da sobre a relação *custo-benefício*. São dados de difícil obtenção em ambientes de acesso automático à informação, sob quaisquer dos paradigmas. Já foi comprovado mesmo, que a própria quantificação dos benefícios de muitas classes de sistemas, muitas vezes, se torna impossível de ser concretizada, impossibilitando a análise de custo-benefício [**SAL 75**, p. 251]. A existência de uma *função de custo* é fundamental para todas as avaliações de eficiência dos sistemas de acesso informacional, em geral, e dos sistemas de RBC, em particular.

A este respeito, Rothenberg, por exemplo, divide os custos que interessam à eficiência e que devem ser avaliados em quatro categorias (In: [**SAL 75**, p. 257]): *(i)* custos inciais de desenvolvimento; *(ii)* custos operacionais de pessoal e material; *(iii)* custos fixos, como aluguel e impostos; e *(iv)* custos na venda dos produtos informacionais resultantes. Desprezando-se este último tipo de custo, o custo de desenvolvimento pode ainda ser sub-dividido em: *(i) design*; *(ii)* teste; *(iii)* operações; e *(iv)* relatórios. Ao mesmo tempo, os custos operacionais podem ser sub-divididos em três partes: *(i)* administrativos; *(ii)* equipamentos; *(iii)* técnicos e profissionais. O resultado será uma função de custo v, expressa como:

$$v = C(t,n) = \sum_{i=1}^{I} f_i(t) + D(t) + O(t,n)$$

onde: t = unidade de tempo; n = unidades operacionais; f_i = a i-ésima subdivisão dos custos fixos por unidade de tempo; $D(t)$ = custos de desenvolvimento, por unidade de tempo, amortizados como custos correntes; $O(t,n)$ = custos operacionais, por unidade de tempo, e por unidade operacional; e $C(t,n)$ = função de custo geral, variando com o tempo e com a quantidade de operações envolvidas.

Na função v os vários fatores ainda podem ser divididos em componentes menores. Assim, os custos operacionais podem ser expressos como:

$$O(t,n) = n(\sum_{j=1}^{J} c_j(t)t_{cj} + \sum_{k=1}^{K} m_k(t)t_{mk} + \sum_{q=1}^{Q} z_q(t)t_{zq})$$

onde: c_j = j-ésima subdivisão dos custos administrativos por unidade de tempo e por unidade operacional (por exemplo, salário do digitador, em certa categoria); m_k = k-ésima subdivisão dos custos de máquina; z_q = q-ésima subdivisão dos custos técnicos; e t_{cj}, t_{mk} , t_{zq} = acréscimos de tempo necessários para as operações administrativas, para as operações de máquina, e para as operações técnicas, respectivamente.

Custos, como estes, costumam ser comparados com a efetividade dos sistemas (*custo-efetividade*) tendo em vista encontrar os mecanismos menos dispendiosos para se realizar um dado conjunto de operações ou quando se pretende maximizar os benefícios de um certo gasto a realizar. O cálculo

destes custos também se torna necessário para efeito de se estabelecer comparação entre eles e os benefícios esperados de um sistema de acesso a bases de casos (*custo-benefício*).

6.9 Conclusão

A avaliação de sistemas baseados em casos é essencial para que se possa entender as fontes de problemas em um sistema já existente ou mesmo para se localizar diferenças fundamentais entre algoritmos e sistemas. Neste capítulo, discutiu-se o enfoque *AVAL*, que oferece uma visão geral destes processos avaliativos com base em idéias já em vigor em outras áreas da computação. No enfoque dado ao problema, foram destacadas as motivações e as razões para se avaliar sistemas de RBC, como ainda os conceitos basilares subjacentes aos projetos de avaliação. Foram identificadas métricas apropriadas à avaliação, com ênfase maior para a avaliação da *efetividade* dos sistemas. Entre as métricas analisadas no capítulo estão as medidas de *retorno*, *precisão*, *retorno médio*, *precisão média*, *refugo*, *retorno de relevância única*, *utilidade*, e a métrica de *eficiência*. Estas são métricas fundamentais para uma avaliação de casos, muito embora outras ainda possam ser propostas, tais como o *quociente de novidade* e o *quociente de cobertura*, de menor significância. O quociente de novidade, por exemplo, representa a razão entre casos relevantes mas ainda não conhecidos por certo usuário e o total de relevantes, de fato, resgatados; enquanto que o quociente de cobertura constitui a razão entre aquele total de casos de fato relevantes resgatados e o total de relevantes apenas previstos antes de um processo de busca.

Parte 3

Experimentos e comparações com pesquisas no mesmo domínio

Como aplicar, concretamente, as metodologias discutidas na Parte 2 desta investigação? Quais as diferenças destas metodologias de RBC em relação a outros paradigmas computacionais já experimentados no mesmo domínio de aplicação? O Capítulo 7 explora estas questões envolvendo a experimentação com esses métodos propostos para o RBC, em um domínio de verdade e não apenas em um "domínio de brinquedo".

Capítulo 7

Experimentos e trabalhos relacionados

Ser similar a ...é apenas algo mais do
que espaços a serem preenchidos.
(Autor desconhecido)

7.1 Introdução

Como experimentar, computacionalmente, com os modelos que acabamos de apresentar? O capítulo aqui introduzido mostra a operacionalização concreta dos modelos desenvolvidos até o presente ponto. Trata-se, por conseguinte, de um capítulo de aplicação e de acoplamentos entre a teoria e a prática, com ênfase para as experimentações de validação dos modelos no domínio particular das decisões sobre crédito. Na aplicação de nossos modelos a este problema particular, no entanto, a meta das experimentações aqui descritas – em nenhum momento – consiste em pretender automatizar via RBC todo o ciclo da *análise empírica* de crédito [IBC ?] necessária para se poder conceder ou não empréstimos financeiros. Nossa meta não consiste, pois, em construir mais um sistema inteligente para o domínio. Antes, porém, a meta por nós estabelecida neste capítulo consiste de duas partes:

- Usar este domínio da *ciência bancária* compreendido pela análise e avaliação de créditos como uma bancada para testar a adequação do domínio e para experimentar com os conceitos, com as fórmulas e com as concepções subjacentes aos modelos *IBT*, *SIM*(*m*,*p*) e *AVAL* de tratamento de casos que funcionem como *respostas* computacionais para o usuário analista.
- Comparar estas nossas metodologias de tratamento de casos com as demais metodologias de tratamento computacional da *análise empírica* do crédito bancário.

Para cumprir esta meta estabelecida, tivemos de enfrentar quatro (4) questões básicas a serem resolvidas nos detalhamentos que seguem:

(i) A questão das razões para a escolha do crédito como um importante domínio de aplicação computacional dos modelos desenvolvidos (na seção 7.2);

(ii) A questão da concepção do modelo de crédito a ser trabalhado computacionalmente. Conceber um tal modelo significa ter de decidir sobre os *atributos* a serem selecionados para caracterizarem créditos ou empréstimos; ou seja, a decisão sobre o *vocabulário* apropriado a casos de crédito a serem indexados. Associada ainda a esta questão do vocabulário para crédito, tivemos de também decidir sobre:

- a *granularidade* destes atributos de crédito;
- a *diagnosticidade* para aqueles atributos de crédito escolhidos para os experimentos;
- a escala de valores para estes atributos; as faixas de votos (denotando faixas de risco por atributos) e limiares de risco por atributo; os níveis de risco por operação de crédito (seção 7.3);

(iii) A questão propriamente da experimentação, a começar pela exibição de casos de crédito com as propriedades estabelecidas em *(ii)* (seções 7.4 e 7.5); e

(iv) A questão, finalmente, da comparação com outras tecnologias computacionais de tratamento do crédito, efetuada na seção 7.6.

7.2 Razões para o domínio do *crédito*

Decisões sobre crédito constituem uma área importante e desafiante para experimentações computacionais por três razões fundamentais: *(i)* crédito é um domínio de fraca teorização, sendo que nele não prevalecem as chamadas *relações causais* ou relações de causa e efeito (*weak domain*); *(ii)* soluções propostas para o domínio de crédito, de fato, também passam a contribuir para uma inteira família de soluções similares de problemas, o que faz aumentar a relevância computacional do domínio; *(iii)* uma solução de RBC para o problema do crédito tem ainda a vantagem de permitir comparações com outras metodologias computacionais já desenvolvidas para o mesmo domínio, conforme analisamos na seqüência.

7.2.1 Decisões sobre crédito baseiam-se em associações

Em análise de crédito, não existe uma teoria a explicar o *modus operandi* do crédito em termos de relações de causa e efeito. Não existe uma *relação causal*, bem definida, entre uma operação de crédito e um cliente que deixa de cumprir um certo contrato. Ao contrário, tudo funciona à base de *associações* entre variáveis [LEW 94]. Ora, este é justamente o tipo de domínio cuja natureza se adequa a um tratamento por RBC. Diferentemente de um modelo causal, sob o enfoque de um *modelo associativo* procura-se associar um ou mais fenômenos da operação de crédito com um ou

mais resultados obtidos ou que se espera obter. Em uma operação de crédito de curto prazo, por exemplo, decisões concretas vão procurar, antes de tudo, predizer um comportamento satisfatório tanto para o tomador quanto para o emprestador. Com este fim, buscar-se-á uma associação entre todos aqueles fatos e atributos que possam concorrer para uma performance de *repagamento* de um crédito. Associar, por exemplo, a área residencial de um cliente (bairro elegante ou favela) com a sua capacidade de repagamento pode oferecer um bom exemplo dos tipos de associações necessárias para nortear a "analise empírica" no domínio do crédito.

7.2.2 Decisões sobre crédito norteiam decisões similares

Metodologias para decisões sobre concessão ou não de créditos também são interessantes por seus efeitos de propagação de usos. Essas metodologias, com as devidas alterações, também podem ser aplicadas a uma inteira família de problemas similares que também costumam ser resolvidos por meio de associação de fatores ou de atributos tais como em [LEW 94]:

- Seleção de pessoal para empregos, dos mais simples aos mais complexos.
- Seleção de solicitantes de cartão de crédito dos mais variados tipos.
- Seleção de impostos e taxas para auditagem visando devoluções, por exemplo.
- Seleção de prisioneiros a serem liberados para programas de trabalho, fora das prisões.

São tarefas para as quais teorias de relações causais não se aplicam, mas onde a associação de atributos descritores de situações e a manipulação de casos podem ser de grande valia.

7.2.3 Decisões sobre crédito permitem comparações metodológicas

Uma solução de RBC para o problema das decisões sobre crédito também se torna interessante porque permite comparações com outras metodologias computacionais já disponíveis operacionalmente para o mesmo problema. Empreendemos, na seção 7.6, uma comparação entre o nosso modelo de decisão sobre crédito com fundamentação no RBC e quatro (04) outras soluções anteriores: *(i)* a solução computacional fundada na metodologia estatística do *credit scoring*; *(ii)* a solução computacional representada pelas redes neuronais; *(iii)* a solução computacional compreendida pelos sistemas baseados em regras; e *(iv)* a solução computacional representada pela aprendizagem de máquina do tipo indutiva compreendida pelo algoritmo ID3.

7.3 Modelo de crédito e modelo de casos a acoplarem-se

Para poder decidir (computacionalmente ou não) sobre qualquer concessão ou sobre qualquer rejei-

ção de um crédito financeiro, um modelo mínimo do universo deste crédito se fará necessário. Concretamente, a nossa experimentação computacional com os modelos de RBC aplicado ao crédito, por conseguinte, vai exigir dois tipos básicos de acoplamentos:

- Primeiro, o acoplamento entre a especificação realística de um *modelo de crédito* e a sua correspondente representação em termos de casos do modelo IBT (teoricamente apresentado no Capítulo 5); e
- Segundo, o acoplamento entre esta representação IBT dos casos de crédito e o modelo $SIM(m,p)$ para a busca da similaridade destes casos de créditos.

Vamos tratar, inicialmente, de especificar o nosso entendimento ou nosso *modelo de operacionalização do crédito* necessário a estes acoplamentos computacionais (seções 7.3.1 a 7.3.6). Ele foi desenvolvido a partir de como o analista de crédito (o *credit underwriter*) costuma enfrentar esta tarefa (e apoiando-se em documento confidencial de instituição financeira ora federalizada). Em seguida, tratamos dos acoplamentos referidos e que constituem a nossa experimentação fundamental (nas seções 7.4 e 7.5). A parte restante do capítulo se dedica às comparações entre as alternativas computacionais para a modelagem de crédito (seção 7.6).

7.3.1 Modelo dos 4 C's do crédito

Analistas tomam em consideração uma série de variáveis normalmente agrupadas nos chamados 4 C's do crédito: *(i)* Caráter do proponente; *(ii)* Capacidade; *(iii)* Capital; *(iv)* Condições. Portanto, são estes os *termos* essenciais que estão na base de um *vocabulário* aplicado às tarefas de análise de crédito. Mais detalhadamente, os fatores a serem considerados na avaliação de um devedor em potencial compreende:

- *Caráter & conceito*. É um fator que expressa uma avaliação do proponente de crédito e de seus avalistas ou fiadores (pessoa física, pessoa jurídica) quanto à sua idoneidade em saldar os compromissos assumidos. Para a formação desse conceito de idoneidade buscar-se-á identificar possíveis desabonos em relação a esse proponente, tais como *(i)* atrasos nas operações já realizadas em bancos; *(ii)* protestos; *(iii)* execuções; *(iv)* falência requerida/decretada e concordata.
- *Capacidade*. É uma variável fundamental para se mensurar, em valores monetários, o limite de crédito do proponente conforme a sua característica. Em relação à pessoas físicas, o exame da capacidade abrange: *(i)* a sua formação e experiência na atividade que exerce; *(ii)* tempo de emprego; *(iii)* empresa em que trabalha, porte e tradição; *(iv)* outros fatores associados à sua capacidade de gerar recursos e assumir compromissos.

- *Capital*. É a avaliação da situação patrimonial do proponente. Envolve o exame de *(i)* rendimento mensal/anual (geração de recursos); *(ii)* relação de bens (acumulação de riquezas decorrentes de sua atividade); *(iii)* endividamentos que recaem sobre os bens ou para a sua manutenção pessoal.
- *Condições*. A avaliação das condições refere-se a fatores alheios à vontade do proponente mas que possam afetar as suas atividades e a sua capacidade de geração de recursos (política governamental, por exemplo).

Como então incorporar esta heurística dos 4 C's – tão comum aos tomadares de decisões sobre crédito bancário – tendo em vista modelar casos de crédito em RBC? A nossa convicção, a este respeito, é a de que os 4 C's sozinhos não servem como atributos de casos computacionais dada a enorme abrangência de cada um deles[2]. Ou seja: embora possam, implicitamente, estar na base da modelagem dos casos, a maior *granularidade* (conseqüentemente, a pouca especificidade ou pouco detalhamento) destes 4 C's inviabiliza o seu emprego direto para efeito de poder indexar ou descrever casos contendo possíveis lições de ajuda nas decisões de crédito.

7.3.2 *Granularidade* orientada para *lições* sobre crédito

Propõe-se então um conjunto de dez (10) atributos básicos e de menor granularidade do que a granularidade expressa nos 4 C's e que possam servir de vocabulário básico na modelagem de casos orientados para lições de crédito. Os atributos selecionados para comporem esses casos de crédito vão descrever:

(i) ora aquelas condições de existência (*ontologia*) das operações de créditos (compreendendo atributos oriundos quer da análise econômico-financeira quer da análise para a decisão, propriamente);

(ii) ora a qualidade dos créditos concedidos ou a *qualidade da decisão* de concedê-los.

Em função deste conjunto de atributos, a fórmula abaixo expressa, então, em que sentido estamos a empregar o conceito computacional de *caso de crédito*:

$$Caso\ de\ Crédito = f\left(\left\{\begin{array}{l} Caráter, \\ Finalidade, \\ Rendimento, \\ Amortização, \\ Capacidade, \\ Documentação, \\ Indicadores, \\ Garantias \\ Risco \end{array}\right\}\right) + Qualidade\ da\ concessão$$

[2] Consulte-se o Anexo B para maiores detalhes sobre o papel da análise de crédito no domínio financeiro.

São estes os sentidos dados aos atributos da fórmula acima:

- *Caráter & Conceito*. Tem o mesmo sentido definido na seção 7.3.1.
- *Finalidade do Crédito ou Financiamento*. Pode se referir a um crédito direto ao consumidor (CDC) para finalidades diversas (como a compra de uma bicicleta) ou pode se referir a um financiamento para empresas.
- *Rendimento*. Significa uma quantificação de todas as rendas mensais/anuais de um cliente.
- *Amortização*. Significa as parcelas de repagamento do crédito contratado, cujos montantes vão depender dos métodos e das bases específicas de cálculo.
- *Capacidade*. Tem o mesmo sentido exposto na seção 7.3.1.
- *Documentação*. Compreende a avaliação da completude e da integridade das informações prestadas em cadastro, por exemplo. No caso de financiamento para empresas, a análise da documentação inclui a qualidade dos demonstrativos contábeis e financeiro; a representação completa e fidedigna dos principais fatos administrativos da empresa.
- *Indicadores financeiros*. Em crédito direto ao consumidor, um indicador útil será dado pelo grau de endividamento do cliente, como também indicadores da capacidade de pagamento. Para as empresas, indicadores úteis incluem os índices de liquidez, os índices de rentabilidade e os índices de solvência, construídos a partir dos balanços.
- *Qualidade e suficiência das garantias*. Leva em conta a liquidez das garantias oferecidas; o valor de liquidação das garantias; nível de depreciabilidade das garantias; grau de controlabilidade das garantias e o custo de liquidação das garantias.
- *Nível de risco da concessão*. Este é um atributo para expressar o *risco* da decisão de conceder um crédito. Trata-se de um *atributo-síntese* de todos os riscos embutidos nos demais atributos.
- *Qualidade da decisão de concessão*. Este é um atributo que estamos a propor para expressar o *resultado avaliativo* da decisão de ter concedido um crédito a um cliente. Ao contrário dos demais atributos que representam todos condições para uma decisão de concessão ou não de um crédito (atributos *ex-ante*, digamos assim), o julgamento da qualidade de uma decisão de concessão constitui um atributo *ex-post*. Um atributo posterior ao ato de emprestar e que, mais diretamente, servirá de *feedback* do sistema para os tomadores de decisão.

Nem todos os casos precisam exibir todos esses dez atributos, mas a presença da maior parte deles

na composição dos casos computacionais fará aumentar a confidência do usuário nos resultados das similaridades a serem computadas.

7.3.3 Escala de valores para atributos do crédito

Os atributos de operações de crédito acima introduzidos vão assumir valores dentro de uma escala a variar de *A* a *D*, com exceção do atributo *Nível de risco* das operações. Nesta escala, cada valor alfabético ou letra representa um grau ou uma caracterização particular de um dado atributo. Esta escala de valores está detalhada no Anexo C, ficando excluído deste anexo apenas a escala de valores para o atributo *Nível de risco* que – dada a necessidade de um método para obtê-la – preferimos analisá-la na seção 7.3.6.1 mais à frente, após a especificação da *diagnosticidade* e a especificação dos votos ou pesos a serem atribuídos a valores destes atributos.

7.3.4 Diagnosticidade dos atributos de crédito

O *modus operandi* sobre como o especialista do domínio julga a importância de cada um destes atributos (e de seus valores) nas operações de concessão de um crédito esta retratado na Figura 7.1. A primeira coluna da figura estabelece a *diagnosticidade* que vamos emprestar a cada atributo selecionado para experimentação, de acordo com o conceito tverskyano já discutido no Capítulo 4 (cf. seção 4.4.4).

Atributos dos Casos	Diagnosticidade	Votos mínimos para valor do atributo	Votos máximos para valor do atributo	Limiar de risco por atributo
1. Caráter & Conceito	2	2	8	5
2. Finalidade	1	1	4	2
3. Rendimento	2	2	8	4
4. Amortização	2	2	8	5
5. Capacidade	2	2	8	6
6. Documentação	1	1	4	2
7. Indicadores financeiros	1	1	4	3
8. Garantias	2	2	8	4
9. Nível de risco	3	*	*	*
10. Qualidade da decisão de concessão	1	1	4	3
Totais de votos e de risco por atributo		**14**	**56**	**34**

Figura 7.1: Importância dos atributos *julgada* pelo analista de crédito

Pesarão – mais significativamente – sobre a concessão (ou não) de um crédito pessoal ou para empresas, por exemplo, os atributos *Caráter*, *Garantias*, e *Nível de Risco* das operação cujas diagnosticidades são 2 e 3 (em uma escala de 1 a 5, por exemplo). Estas diagnosticidades são superiores às

dos atributos *Finalidade* do empréstimo, *Indicadores financeiros*, e até mesmo do atributo *Documentação* exigida na operação (tudo a depender, porém, da *política de créditos* que venha a ser adotada por uma instituição creditícia).

7.3.5 Faixas de votos ou risco e limiar de risco por atributo

Além das diagnosticidades dos atributos e além da escala de valores estabelecida para os atributos (de A a F e de A a D, como descritas nas seções anteriores e no Anexo C), também está sendo adotada uma faixa de importância mínima e importância máxima para os valores de cada atributo, conforme consta nas colunas 2 e 3 da mesma Figura 7.1. Pode-se pensar estas importâncias como sendo votos dados aos valores dos atributos em *função dos riscos* destes valores. Ou seja: *faixa de votos mínimos* e *faixa de votos máximos* para valores de atributos correspondem, em nossa modelagem, a faixas de risco mínimo e máximo (na linguagem dos analistas de crédito).

A coluna 4 e última da Figura 7.1 introduz também uma outra propriedade na modelagem de nossos casos de crédito: somente poderão ser permitidas operações de crédito que se situem nos patamares de risco máximo estabelecidos para cada atributo. Esta condição dá origem a um *limiar total de risco* de 34 pontos (correspondente ao somatório dos limiares de risco/votos particulares para cada atributo da operação de crédito, conforme mostra a linha 11 da Figura 7.1.

Observe-se na Figura 7.1 o papel dos asteriscos (*). Eles indicam que as quantidades mínimas e máximas de votos a serem dados ao valor do atributo *Nível de risco* vão ser determinadas na seção seguinte. Ou seja: as importâncias para o atributo *Nível de risco* ainda irão depender dos totais mostrados na linha 11 da mesma figura, conforme discussão que segue.

7.3.6 Níveis de risco admitidos por operação

Os votos de risco atribuídos a cada atributo (Figura 7.1 – colunas 2, 3, e 4) vão definir o nível de risco de uma operação, como um todo. Isto significa que tanto o *total* de votos mínimos para o valor de cada atributo (14, na coluna 2) quanto o total dos votos máximos (56, na coluna 3) quanto ainda o *limiar total de risco* dos atributos (34, na coluna 4) irão determinar os valores para aquele que é o mais importante dos atributos, qual seja, o *Nível de risco* de um crédito como um todo.

7.3.6.1 Valores e pesos do atributo *Nível de risco*

A Figura 7.2 especifica estes valores. Nela, são mostradas três especificações das operações de crédito ora em modelagem: *(i)* os valores atribuídos a este atributo (de A até F); *(ii)* as faixas de votos de importância para estes valores assumidos pelo atributo *Nível de risco*; *(iii)* como ainda o

significado destes riscos classificados em termos destas faixas.

NÍVEL DE RISCO (VALORES)	PESOS PARA VALORES (DO NÍVEL DE RISCO)	SIGNIFICADO DOS INTERVALOS DE RISCO
Nível 1 = A	14 – 20	Risco inexistente ou muito baixo. Ótimas perspectivas de sucesso do empréstimo/financiamento.
Nível 2 = B	21 – 27	Risco mínimo ou moderado, abaixo do prevalecente para os empréstimos da área. Boas perspectivas de sucesso do empréstimo/financiamento.
Nível 3 = C	28 – 34	Risco aparente médio e tolerável, merecendo acompanhamento próximo.
Nível 4 = D	35 – 41	Risco presente e ligeiramente acima da média das operações, requerendo um maior acompanhamento no repagamento do crédito concedido.
Nível 5 = E	42 – 48	Risco elevado, em relação à carteira do banco e ao setor empresarial.
Nível 6 = F	49 – 56	Risco total; potencialmente muito elevado, situando-se na área mais crítica, com alta probabilidade de ocorrência de problemas no cumprimento das obrigações contratuais junto ao emprestador.

Figura 7.2: Atributo *Nível de Risco* e sua qualificação

7.3.6.2 Obtenção do atributo *Nível de risco* e de seus intervalos

Note-se que os valores para o atributo *Nível de risco* são determinados em função da quantidade de pesos ou votos que venham a ser atribuídos ao risco de uma determinada operação de crédito, em particular. Esta mensuração de níveis de risco de cada operação está especificada na Figura 7.2 em termos de intervalos de votos, dando origem a uma escala de valores que varia de A a F.

Observem-se ainda as correlações entre as Figuras 7.2 e 7.1. No risco de *Nível 1* ou risco de valor A (na Figura 7.2), a quantidade de votos ou pesos atribuídos a este valor A varia de 14 a 20, sendo 14, portanto, o limite inferior do primeiro intervalo de pesos. Por outro lado, o último intervalo de votos para valores do atributo *Nível de risco* tem como limite superior a quantidade de 56 votos. Isto corresponde, justamente, ao total de votos mínimos e ao total de votos máximos atribuídos a todos os demais atributos, e que foram especificados na linha 11 da Figura 7.1. Estamos trabalhando com a restrição de que a instituição creditícia somente admitirá negócios com o risco máximo de 34 (trinta e quatro pontos) que definem o *Nível 3* (valor C) da Figura 7.2. Ora, a quantidade 34, sendo o limite superior do intervalo definidor do *Nível 3*, corresponde, justamente, ao piso total de

risco dos atributos também dado na linha 11 da Figura 7.1. Daí o fato de somente serem admitidos negócios com risco de nível C.

São estes níveis de risco e respectivos pesos que devem ser os componentes principais a serem incorporados aos casos de crédito, na forma IB-T de nosso interesse.

7.4 Casos de *Crédito*: exemplos na forma IBT

7.4.1 Forma genérica dos casos de crédito

Havendo, assim, descrito os conceitos embutidos no modelo de crédito por nós construido para acoplar-se ao nosso modelo de caso, a Figura 7.3 ilustra, agora, o primeiro destes acoplamentos que dá origem à *forma genérica* das operações de crédito, em termos de casos do modelo IBT.

Caso 000		
Diagnosticidade-Atributo	**Atributo:Valor**	**Votos-valor-atributo**
2	*Concessão do Crédito* Caráter: Valor-1	Votos-1
1	Finalidade: Valor-2	Votos-2
2	Rendimento:Valor-3	Votos-3
2	Amortização:Valor-4	Votos-4
2	Capacidade:Valor-5	Votos-5
1	Documentação:Valor-6	Votos-6
1	Indicadores:Valor-7	Votos-7
2	Garantias:Valor-8	Votos-8
3	Nível de risco:Valor-9	Votos-9
1	*Resultado da Concessão* Qualidade da decisão:Valor-10	Votos-10

Figura 7.3: *Caso genérico* de crédito como objeto tverskyano

Tudo que foi estabelecido nas seções anteriores a respeito da modelagem e caracterização das operações de um crédito cabe no interior dos casos da forma mostrada na Figura 7.3.

7.4.2 Instâncias de casos de crédito

As Figuras 7.4 (*Casos* X1 e X2), 7.5 (*Caso* 003), 7.6 (*Caso* 004) e 7.7 (*Caso* 005) mostram, por sua vez, instanciações particulares do modelo computacional genérico de crédito; são exemplos de indexação de casos de crédito – para efeito de se poder examinar o segundo e mais importante tipo de acoplamento já referido. Ou seja, aquele acoplamento entre o modelo IBT – aplicado a créditos financeiros – e o modelo de mensuração da similaridade das decisões tomadas sobre crédito.

Nossa experimentação tem lugar neste ponto, justamente, para viabilizar e validar estes acopla-

mentos entre: IBT e *SIM*(*m*,*p*), *SIM*(*m*,*p*) e *ORDEN*, e entre esses modelos computacionais e o modelo de crédito descrito. A experimentação, fundamentalmente, consiste na programação destes modelos sem a ajuda de qualquer ferramenta de RBC. Outras informações sobre sessões de busca de similaridades estão apresentadas no Anexo D.

7.5 Experimentação computacional: resultados básicos

Vamos tomar, aleatoriamente, casos de crédito de uma dada base de casos de empréstimo[3], tendo em vista, antes de tudo, a aplicação da nova métrica de similaridade já formulada. Sobretudo, a nossa meta nesta experimentação realizada teve como objetivos:

- Mostrar, passo a passo, a aplicação prática da nova métrica *SIM*(*m, p*) (seção 7.5.1);
- Discutir aqueles aspectos mais relevantes de tomada de decisão (por parte do analista de crédito) decorrentes da similaridade encontrada para os casos considerados para comparações (seção 7.5.2);
- Comparar, de um modo sintético, os resultados obtidos pela aplicação da nova métrica com resultados obtidos por uma outra popular métrica de similaridade (seção 7.5.3).

Tomemos, por exemplo, o *Caso* 003. O conhecimento embutido no interior deste caso representa uma tomada de decisão de sucesso (onde o agente emprestador provavelmente teve um grande lucro dada a alta qualidade da decisão). Enquanto isto, o *Caso* 004, por sua vez, retrata uma situação de insucesso (o agente emprestador teve prejuízos dado a uma decisão problemática). Também, o *Caso* 005 representa uma situação de sucesso, mesmo não tendo sido registrado no interior do caso o atributo *Finalidade* do referido empréstimo. Por fim, os *Casos* X1 e X2 que são *casos-indagação*. Ou seja: casos-indagação são aqueles casos em mãos do usuário a demandar um outro da base de casos para efeito de comparações e de busca de similaridade (no *Caso* X1, apenas um atributo está sendo desconhecido pelo usuário, indicado pelo símbolo "?"; enquanto isto, no *Caso* X2 dois dos atributos estão ausentes do julgamento do analista).

7.5.1 Aplicação da métrica nova

Qual de entre o *Caso* 003 (Figura 7.5) e o *Caso* 004 (Figura 7.6) será o mais similar ao novo *Caso* X1 do usuário (Figura 7.4)?

[3] Um conjunto de casos-semente para a experimentação com uma base de casos de crédito encontra-se no anexo E.

Caso X1		
Diagnosticidade-Atributo	Atributo:Valor	Votos-valor-atributo
	Concessão do Crédito	
2	Caráter: A	**2**
1	Finalidade: B	2
2	Rendimento: D	4
2	Amortização: C	6
2	Capacidade: A	2
1	Documentação: B	2
1	Indicadores: B	2
2	Garantias: D	6
3	Nível de risco: A	18
	Resultado da Concessão	
1	Qualidade da decisão: ?	**?**

Caso X2		
Diagnosticidade-Atributo	Atributo:Valor	Votos-valor-atributo
	Concessão do Crédito	
2	Caráter: A	**2**
1	Finalidade: B	2
2	Rendimento: D	4
2	Amortização: C	6
2	Capacidade: ?	?
1	Documentação: B	2
1	Indicadores: B	2
2	Garantias: D	6
3	Nível de risco: A	18
	Resultado da Concessão	
1	Qualidade da decisão: ?	**?**

Figura 7.4: Casos de crédito ainda a serem concedidos

Caso 003		
Diagnosticidade-Atributo	Atributo:Valor	Votos-valor-atributo
	Concessão do Crédito	
2	Caráter: D	5
1	Finalidade: B	2
2	Rendimento: D	4
2	Amortização: A	5
2	Capacidade: D	6
1	Documentação: B	2
1	Indicadores: C	3
2	Garantias: D	4
3	Nível de risco: A	14
	Resultado da Concessão	
1	Qualidade da decisão: D	**2**

Figura 7.5: Caso de crédito com insucesso na base de casos

Caso 004		
Diagnosticidade-Atributo	**Atributo:Valor**	**Votos-valor-atributo**
	Concessão do Crédito	
2	Caráter: D	5
1	Finalidade: A	2
2	Rendimento: D	4
2	Amortização: A	4
2	Capacidade: C	6
1	Documentação: B	2
1	Indicadores: C	3
2	Garantias: C	4
3	Nível de risco: A	14
	Resultado da Concessão	
1	Qualidade da decisão: B	3

Figura 7.6: Caso de crédito com sucesso, na base de casos

Caso 005		
Diagnosticidade-Atributo	**Atributo:Valor**	**Votos-valor-atributo**
	Concessão do Crédito	
2	Caráter: D	5
1	Finalidade: ?	?
2	Rendimento: D	4
2	Amortização: A	5
2	Capacidade: C	6
1	Documentação: B	2
1	Indicadores: C	3
2	Garantias: C	4
3	Nível de risco: A	14
	Resultado da Concessão	
1	Qualidade da decisão: D	3

Figura 7.7: Caso de crédito com sucesso e ausência do atributo *Finalidade*

Caso X1 comparado com *Caso* 003

Aplicando o nosso modelo $SIM(m,p)$ de similaridade, toma-se p como sendo o *Caso* X1 do usuário para o qual se busca similares e m como sendo um dos casos da memória – por exemplo – o *Caso* 003. Ou seja, para computar a similaridade dada por *SIM*(*Caso* 003, *Caso* X) – primeiramente determina-se: *(i)* o conjunto daqueles atributos comuns aos dois casos; e em seguida *(ii)* o conjunto dos atributos *disjuntos* ou atributos não compartilhados pelos casos em consideração, nos termos da seção 4.5.1 do Capítulo 4. Estes conjuntos são:

- $M \cap P$: *Caso* 003 $\cap$ *Caso* X1 = {*Caráter, Finalidade, Rendimento, Amortização, Capacidade, Documentação, Indicadores, Garantias, Nível de risco*}
- $M - P$: *Caso* 003 - *Caso* X1 = {*Qualidade da decisão*}
- $P - M$: *Caso* X1 - *Caso* 003 = $\varnothing$

Considerando-se agora a quantidade de votos ou pontos atribuídos aos atributos de cada conjunto acima e também aplicando-se a métrica *SIM*(*m,p*) que foi estendida a partir da métrica tverskyana (seção 4.5.1) obtem-se então a seguinte expressão:

$$\begin{aligned} SIM(Caso\ X1, Caso\ 003) = {} & a\times(2\times2\times0.64 + 1\times2\times1 + 2\times4\times1 + 2\times5\times0.88 + 2\times2\times0.52 + 1\times2 \\ & \times1 + 1\times2\times0.88 + 2\times4\times0.76 + 3\times14\times 0.3) \\ & - b\times(1\times2) - c\times(0) \\ = {} & 147\times a - 2\times b - 0\times c. \end{aligned}$$

Estamos designando a expressão $147\times a - 2\times b - 0\times c$ de *pré-processamento da similaridade* entre créditos (cf. seções 4.5.2 e 4.5.3), justamente porque o valor final da similaridade, de algum modo, ainda vai depender dos valores a serem atribuídos aos parâmetros envolvidos, quer pelo sistema quer pelo usuário. Para os seguintes valores de parâmetros $a = 3$; $b = 1$; $c = 1$, nós obtemos a similaridade dada por *SIM*(*Caso* X1, *Caso* 003) = 145.

Observe-se aqui a maneira como foram computados aqueles votos dados aos atributos *compartilhados* pelos casos – diferentemente da computação dos votos aos atributos *não compartilhados* – de acordo com a formulação de $\underline{w}$ (da seção 4.5.1). Ou seja: 2 votos para o valor de *Caráter*, no *Caso* X1, em vez de 5 votos do *Caso* 003; 5 votos para o valor de *Amortização*, no *Caso* 003, em vez de 6 votos do *Caso* X1; 14 votos para o valor do atributo *Nível de risco*, no *Caso* 003, em vez do valor 18 para *Nível de risco* do *Caso* X1, devendo ainda cada um destes valores mínimos ser multiplicado por seus respectivos fatores de correção.

Caso X1 comparado com *Caso* 004

A determinação da similaridade com quaisquer outros casos de crédito segue a mesma metodologia acima. Coincidentemente, os conjuntos de atributos de casos para a similaridade *SIM*(*Caso* X1, *Caso* 004,) também serão os mesmos do exemplo anterior:

- $M \cap P$: *Caso* 004 $\cap$ *Caso* X1 = {*Caráter, Finalidade, Rendimento, Amortização, Capacidade, Documentação, Indicadores, Garantias, Nível de risco*}
- $M - P$: *Caso* 004 - *Caso* X1 = {*Qualidade de Decisão*}
- $P - M$: *Caso* X1 - *Caso* 004 = $\varnothing$

Mesmo com esta coincidência da presença dos mesmos atributos nos dois casos dados, o cálculo realizado vai exibir um diferente resultado dado por: *SIM*(*Caso* X1, *Caso* 004) = 154 (considerando-se os mesmos valores para os parâmetros *a*, *b*, *c*, do exemplo anterior).

Considere-se, adicionalmente, os *Casos* 005, 017 e 019 também presentes na base de casos (Ver o

Anexo E). A aplicação da mesma métrica proposta leva às seguintes computações e às ordenações de casos apresentadas na Figura 7.8.

Rank dos casos	Identificação dos casos	*SIM(m,p)*
3	003	145
1	004	154
2	005	146
4	017	142
5	019	98

Figura 7.8: *Ranking* utilizando a métrica proposta

7.5.2 Tomadas de decisão guiadas pela similaridade

Que lições extrair deste quadro de similaridades de casos de operações de crédito? As *lições* contidas nesses casos representantes de operações de crédito podem ser úteis ao usuário analista de crédito de duas maneiras principais:

- quer em relação ao desfecho final das decisões passadas sobre empréstimos (desfecho representado pelo atributo *Qualidade da decisão*);
- quer em relação a algum outro atributo, valores, pesos ou diagnosticidades que podem servir para comparações com uma nova solicitação a ser decidida.

Como visto na Figura 7.8 o caso mais similar ao *Caso* X1, é o *Caso* 004, pois, quanto maior for o valor final de *SIM(m,p)* mais próxima será a similaridade entre os casos comparados e vice-versa. Para o usuário, isto significa que a aplicação de nosso modelo recomendará que a nova solicitação X1 de empréstimo deva ser aceita pelo analista (dada a similaridade com um caso de boa qualidade na decisão). Contrariamente, se o caso mais próximo de X1 tivesse sido o *Caso* 003, nesta situação a similaridade estaria a recomendar uma não aceitação da solicitação de crédito, considerando-se que o *Caso* 003 constitui um exemplo de crédito que no passado criou problemas para a organização creditícia. Os resultados da comparação computacional mostram, portanto, que o *Caso* 004 contém uma lição sobre uma *Qualidade de decisão* já tomada e de valor B, em oposição à *Qualidade de decisão* de valor D, no *Caso* 003.

Captação de diferenças mínimas de indexação

Observe-se que os *Casos* 003 e 004 foram indexados de um modo muito parecido. Houve maiores divergências entre os valores dos atributos (por exemplo, *Finalidade* B no *Caso* 003 e *Finalidade*

A no *Caso* 004, etc). Porém, a lógica de "pontagem" para os valores, em ambos os casos, foi praticamente a mesma. A única diferença verificada entre votos para valores de atributos foi em relação ao atributo *Qualidade da decisão* (3 pontos ou votos para o valor B deste atributo, no *Caso* 004 contra 2 votos para o valor D, no *Caso* 003). Mesmo assim, a métrica de similaridade $SIM(m,p)$ foi suficientemente eficaz para captar esta pequena diferenciação na "pontagem" em votos dos valores de atributo. Como resultado da comparação foi exibido o *Caso* 004 como o mais similar, trazendo para o usuário uma lição de decisão qualitativamente boa a merecer a consideração do usuário.

Tolerância à ausência de atributos

Considere-se agora o comportamento da métrica $SIM(m,p)$ em relação à ausência de atributos no interior dos casos cuja similaridade esteja sendo mensurada. No *Caso* 005 apenas o atributo *Finalidade* está ausente, assim como no *Caso* X1 apenas o atributo *Qualidade da decisão* está ausente; enquanto isto, o *Caso* X2 deixa de representar (por alguma razão) dois atributos: *Capacidade* e *Qualidade da Decisão*. A métrica também vai tolerar esta ausência de atributos, de tal modo a degradar a mensuração da similaridade de uma maneira muito suave. Por exemplo, no *Caso* 005 e no *Caso X1*, um total de dois atributos estão ausentes mas pode alcançar uma similaridade SIM(*Caso* X1, *Caso* 005) = 146. Ou seja: na falta de uma maior quantidade de atributos, a mensuração da similaridade certamente vai decrescer; porém, sem apresentar movimentos de queda brusca de valores, em relação às similaridades aqui trabalhadas.

7.5.3 Métrica nova × métrica *Weighted Block-City*

Nossa investigação, antes de enfatizar a comparação entre os resultados produzidos por métricas de similaridade, procura enfatizar a comparação entre as metodologias existentes para o tratamento computacional da análise de crédito. Esta será a problemática a ser examinada na próxima seção.

Mesmo, assim, podemos estabelecer comparações entre a métrica por nós proposta e uma outra métrica alternativa tal como a denominada métrica *Weighted Block-City*, exibida no Anexo A. Considerando-se os mesmos casos tomados quando da aplicação da métrica nova acima, a aplicação da *Weighted Block-City* a esses casos é capaz de dar origem às computações e ordenações de casos apresentadas na Figura 7.9.

Rank dos casos	Identificação dos casos	*Similaridade Weighted Block-City*
1	004	16
1	003	16
2	005	15
3	017	14
4	019	11

Figura 7.9: *Ranking* de casos utilizando uma métrica alternativa

Os resultados comparativos entre estas duas medições de similaridade podem ser detalhados como seguem:

Resultado 1: Observe-se que os valores obtidos para as similaridades, tanto usando a métrica nova quanto empregando a *weighted block-city* não divergem bastante ou são valores próximos entre si, dentro de cada métrica, respectivamente. Isto pode expressar, segundo J. Kolodner [**KOL 93**, p. 357], duas realidades: Ou as similaridades entre os casos, de fato, existem e foram captadas, tanto em uma como na outra métrica, ou então os critérios de indexação/ponderação dos atributos levaram a este resultado de proximidade de valores. Nossa convicção é a de que ambos os fatores concorreram para esta tal proximidade.

Resultado 2: Observe-se também que há uma coincidência entre os *rankings* dos *Casos* 04 e 19 que são o primeiro e o último caso, nas duas metodologias.

Resultado 3: As diferenças metodológicas, porém, começam a aparecer quando se considera os valores intermediários das Figuras 7.8 e 7.9, certamente, dado ao fato de que a *weighted block-city* é incapaz de lidar com elementos qualitativos tais como os valores assumidos por atributos. Esta métrica considera apenas a presença ou não de um certo atributo em um dado caso.

Resultado 4: Por não considerar os valores assumidos por atributos a métrica *weighted block-city* foi incapaz de captar a natureza dos *Casos* 004 e 003 que são casos de empréstimos bom e ruim respectivamente, grupando-os no mesmo *rank*. A nossa métrica foi capaz de jogar o *Caso* 004 na primeira posição e o *Caso* 003 na terceira posição.

7.6 Comparações com outras metodologias de crédito

Foram discutidos acima resultados implementacionais dos modelos propostos. Torna-se interessante, neste ponto, comparar estes resultados descritos com outras tecnologias de tratamento computacional do crédito financeiro. Vamos estabelecer comparações com outras abordagens automáticas para o crédito e que constituem metodologias alternativas ao tratamento baseado em casos por nós investigado. Porém, as comparações aqui efetuadas vão se referir à natureza destas metodologias alternativas de crédito e não, propriamente, a quaisquer resultados de suas implementações. Realizar comparações metodológicas, do ponto de vista dos resultados de implementação, constitui um problema que foge ao escopo das nossas investigações. Examinamos, pois, na seqüência estas metodologias a começar pelo *credit scoring*.

7.6.1 Comparações com o *credit scoring*

Credit scoring é uma ferramenta estatística para estimar riscos; no caso em apreço, o risco de que um tomador de empréstimos venha a deixar de cumprir os seus compromissos de repagamento. O enorme volume das operações de crédito, o acréscimo da demanda pelo crédito ao consumidor e o rápido crescimento da sociedade de consumo levaram ao emprego desta ferramenta em substituição ao julgamento individual por parte do analista de empréstimos. O julgamento do tipo individual em decisões de crédito é problemático por duas razões básicas [LEW 94]:

(i) introduz *inconsistências* nas decisões de crédito, ao longo do tempo, e

(ii) introduz *não uniformidade* de decisões entre grupos de analistas de crédito.

Ora, inconsistência e não uniformidade de tratamento são problemas muito graves em paises de capitalismo avançado, com legislações/regulações severas em relação a possíveis discriminações pessoais nas decisões de crédito. Um solicitante de crédito que teve rejeitada a sua proposta – se vier a confrontá-la com alguma outra similar que já tenha sido aprovada – poderá acionar a legislação contra a discriminação na concessão, o que implicará em danos jurídicos para qualquer agente financeiro (no caso norte-americano, por exemplo). Daí o interesse em evitar os julgamentos pessoais e buscar a automação do processo de decisão sobre o crédito através do *credit scoring*, entre outros mecanismos.

7.6.1.1 Tratamento estatístico do crédito

Suponha que tenhamos um conjunto de exemplos de empréstimos já decididos (um banco de dados dos créditos passados). Suponha também que seja selecionado um certo *atributo alvo* C_l nos exemplos de crédito desta base de dados, com $l = 1, ..., L$. Este atributo alvo C_l (e o respectivo valor por

ele assumido) é também conhecido por *classe*. O tratamento estatístico do *credit scoring*, fundamentalmente, objetiva avaliar a probabilidade Prob(C_l | X) de que uma nova proposta X de crédito venha a pertencer a uma classe C_l, em relação às outras classes dos exemplos em sua vizinhança (por exemplo, pertencer à classe dos exemplos passados de crédito que não causaram quaisquer problemas jurídicos, dada a alta *qualidade da decisão* tomada).

Esta probabilidade de que uma nova solicitação de crédito venha a pertencer a uma classe previamente selecionada pode ser achada pelo teorema de Bayes:

$$\text{Prob}(C_l \mid X) = \frac{f_l(X)\text{Prob}(C_l)}{f(X)}$$

Onde: Prob(C_l) é a probabilidade *a priori* da classe C_l em uma base de exemplos de crédito; $f_l(X)$ é a função densidade da classe C_l na vizinhança de X; f(X) é a função de densidade da base de exemplos, na vizinhança de X.

Operacionalmente, este tamanho da vizinhança é dado pela quantidade dos exemplos que a ela pertençam (isto é, dado por *k* operações de crédito constantes na memória de exemplos) e as funções de densidade são estimadas em termos de freqüências. Essa probabilidade é então simplificada para:

$$\text{Prob}(C_l \mid X) \cong n_l/k$$

onde n_l é a quantidade de exemplos, dentre as *k* operações de crédito, que pertençam à classe C_l. O novo empréstimo X ficará associado àquela classe C_l que tiver a mais alta probabilidade.

7.6.1.2 *Credit scoring* × casos de crédito

Em relação aos nossos modelos, observa-se que:

- *Credit scoring* baseado em probabilidade e casos de crédito têm, fundamentalmente, uma propriedade em comum: envolvem comparações de situações passadas (créditos passados) com situação corrente (crédito proposto).
- Ambas as metodologias também possuem a característica de trabalharem com atributos de crédito e seus valores. Porém, enquanto os atributos, em *credit scoring*, servem para a computação de escores a originarem curvas de densidade, em RBC, por outro lado, esses atributos e valores servem para a modelagem de casos cujas similaridades vão ser computadas.
- *Credit scoring* assim como casos de crédito tanto pode ser implementado através de ferramentas ou *shells* como pode ser diretamente programado através de linguagens apropri-

adas à manipulação estatística tal como a linguagem SAS.

- *Credit scoring* funciona bem com grandes quantidades de dados padronizados para testar hipóteses conhecidas. Contudo, grande parte dos métodos estatísticos é pouco apropriada para análise exploratória (ou seja, quando as hipóteses não sejam ainda bem conhecidas). A metodologia também parece mais vulnerável quando atributos aparecem cujos valores são desconhecidos (podem existir e serem desconhecidos ou podem simplesmente ser nulos).
- *Credit scoring,* por fim, dificilmente leva em conta o conhecimento de senso comum ou conhecimento de *background.* Em RBC esse conhecimento torna-se representável através da integração de técnicas numéricas e simbólicas disponíveis nos métodos de indexação.

7.6.2 Comparações com redes neuronais

Não nos foi possível identificar abordagens de redes neuronais, especificamente, voltadas para o domínio do crédito. Em termos teóricos, no entanto, redes neuronais funcionam até melhor do que o RBC em ambientes: *(i)* caracterizados por conhecimentos escassos e esparsos, e *(ii)* quando os dados não possam ser facilmente representados simbolicamente, tal como em reconhecimento de sinais de radar, reconhecimento de padrões, visão computacional, fala e processamento de imagens.

Redes neuronais são bastante resistentes a *ruídos* durante a fase de consulta: por exemplo, mesmo quando somente uma fração dos atributos originais venham a assumir valores, a performance do resgate pode ainda ser muito alta. Neste sentido, redes neuronais em crédito pode se constituir uma proposta promissora de investigação. A própria metodologia de tratamento de pesos – o que é um aspecto forte em redes neuronais – pode se converter em um fator favorável a uma abordagem do crédito financeiro por redes neuronais.

Redes neuronais, porém, apresentam problemas [ALT 95]: Primeiro, redes não são apropriadas quando se tem de considerar o conhecimento de *background.* Segundo, redes não podem lidar com complexos dados estruturados. Terceiro, ergonomicamente, o paradigma de redes neuronais sofre da falta de transparência. Usuários não podem julgar a validade das decisões da rede dada a própria natureza do seu funcionamento interno: a *saída* da rede é uma função de vetores ponderados que depende da arquitetura da rede e do tipo de aprendizagem empregado. Qualquer explicação ou justificação desta *saída* é difícil de ser conseguida.

7.6.3 Comparações com os *sistemas baseados em regras*

Também *sistemas baseados em regras* heurísticas (sistemas expertos) têm sido experimentados no

tratamento computacional de empréstimos financeiros sendo o sistema CLUES [**TAL 95**] o mais interessante deles tanto do ponto de vista do binômio *IA* × *Análise de Crédito* quanto do ponto de vista das comparações com os nossos modelos. A arquitetura CLUES foi descrita – de um modo ainda muito geral – por Houman Talebzadeck e seu grupo, em recente *AI Magazine* (Spring, 1995). Nela, adota-se de antemão, o postulado de que a análise de crédito não constitui propriamente um processo sistemático; isto significa que a tomada de decisão sobre crédito, por natureza, constitui um processo que não adota uma abordagem passo a passo ou algorítmica. O analista, pelo contrário, tem de examinar cada componente de uma solicitação de crédito, individualmente, para poder descobrir cada ponto forte e também cada ponto fraco de uma solicitação de crédito e assim avaliar como estes pontos se influenciam mutuamente. É um processo sobretudo intuitivo, em vez de científico [**TAL 95**].

7.6.3.1 Arquitetura CLUES

Com base nestes pressupostos, os criadores do sistema CLUES dividiram as análises automáticas a serem empreendidas para um dado empréstimo em três categorias:

- *Análise de crédito*. Tendo em vista examinar o histórico creditício do tomador;
- *Análise de capacidade*. Tendo em vista determinar a capacidade de amortização mensal do empréstimo por parte do tomador;
- *Análise avaliativa ou apreciativa*. Tendo em vista a integração daqueles dados fundamentais das duas análises anteriores para efeito da decisão final.

Estas divisões da tarefa de análise global deram origem então a uma arquitetura com três módulos automáticos fundamentais, cada um deles correspondendo a cada análise particular. Estes módulos da arquitetura CLUES são mostrados na Figura 7.10.

Estes três módulos principais são ainda complementados por dois outros:

(i) um módulo que examina a completude dos dados necessários à tomada de decisão pelo sistema; e

(ii) um módulo responsável pelos cálculos envolvidos em uma operação de empréstimo.

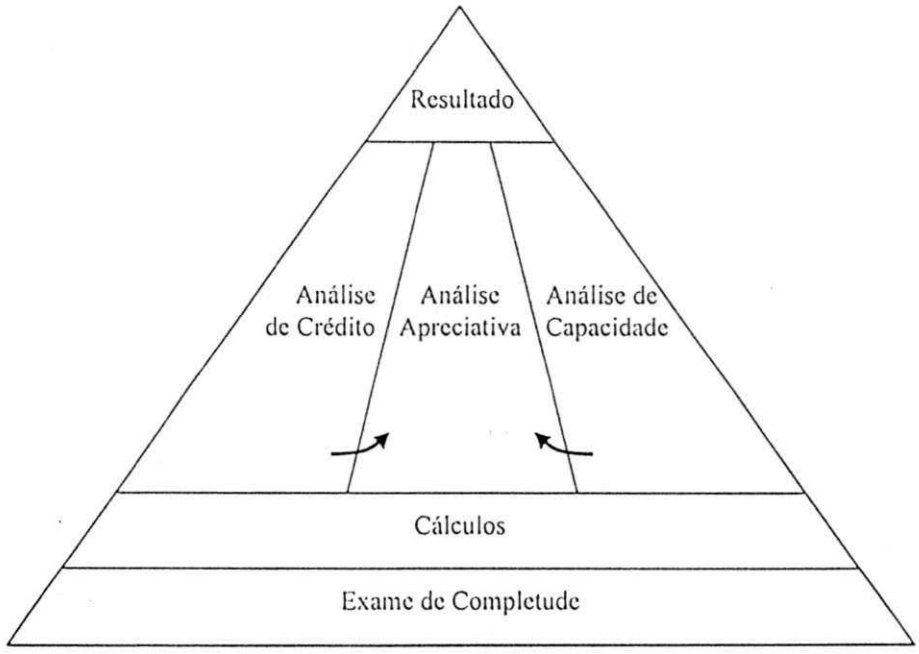

Figura 7.10: Módulos da arquitetura CLUES

Funcionamento de regras computacionais para crédito

Regras específicas para cada um destes módulos foram criadas tendo em vista a sua aplicação na recomendação ou não de um crédito particular. Regras foram desenvolvidas, por exemplo, para avaliar informações tais como: *renda*, *emprego*, *patrimônio*, *disponibilidades* e *risco do empréstimo*. Baseado no resultado de cada um destes módulos formados por regras, o sistema alcança então uma decisão para conceder ou não um empréstimo. Cada módulo pode afetar positiva ou negativamente os resultados de um outro módulo. Se a *relação empregatícia* de um tomador, por exemplo, for considerada pelo sistema como sendo uma relação de emprego instável, este fato será problemático para a avaliação da renda do tomador. Aquela informação sobre a renda advinda deste emprego instável não será repassada para aquele outro conjunto de regras (aquele módulo) responsável pela análise de renda (o que vem a reduzir as chances de um tomador). O contrário também é verdadeiro; se o sistema determinar que a renda do tomador é menor do que a desejável, porém, este tomador possui um sólido patrimônio de liquidez ou semi-liquidez imediata então o risco do empréstimo pode ser significativamente reduzido. Esta redução de risco contará pontos para uma decisão de empréstimo.

Volatilidade das regras e do mercado

Aproximadamente 1000 regras compõem o sistema CLUES. De onde provêm estas regras? Modeladas pelo projetista, a maior parte destas regras tem como origem as próprias *diretrizes internas*

emanadas de cada organismo financeiro interessado como também as diretrizes emanadas das autoridades monetárias. Essas diretrizes internas, por sua vez, retratam as situações momentâneas do mercado financeiro. Ora, o mercado de dinheiro é sinônimo de *volatilidade* – uma propriedade perturbadora para qualquer modelo de regras. Ao alterar-se o "humor" do mercado de dinheiro, alterar-se-ão também as regras modeladoras desse mercado e, conseqüentemente, também tanto a necessidade quanto o ritmo de manutenção deste sistema.

7.6.3.2 Regras de crédito × Casos de crédito

Esta abordagem computacional do crédito através de regras, claramente, contrasta com o nosso tratamento baseado em casos de crédito, conforme as diferenças abaixo apontadas.

- Regras para decisões de crédito: são impróprias porque o crédito constitui um domínio instável, por natureza. Casos de crédito, ao contrário, representam experiências de decisões já tomadas e absorvem melhor a volatilidade do domínio.
- Regras para decisões de crédito: funcionam como *padrões*. Casos de crédito funcionam como *constantes*.
- Regras para decisões de crédito: como conseqüência da volatilidade do domínio necessitam de constante manutenção da base de conhecimento. Casos de crédito necessitam apenas da manutenção comum aos sistemas.
- Regras para decisões de crédito: são pequenas e, idealmente, independentes porções de conhecimento sobre crédito. Constantes modificações destas regras (também voláteis) podem introduzir inconsistências no sistema. Casos de crédito podem representar porções maiores ou menores de conhecimento (*cache*), sem esta preocupação de inconsistências por volatilidade.

7.6.4 Comparações com a *aprendizagem indutiva* do tipo *ID3*

As conexões entre o RBC e a indução (via árvore de discriminação e algoritmo ID3) já foram analisadas ao se tratar da armazenagem e do resgate de casos no Capítulo 2 sobre o estado da arte do RBC (seções 2.6.2 e 2.7.6). Interessa-nos examinar, neste ponto de nossa investigação, sobretudo o aspecto da aplicação do algoritmo ID3 ao domínio do crédito financeiro, particularmente. Dispomos apenas de uma vaga informação sobre esta aplicação advinda do Japão, muito embora seja ela uma informação bem atualizada, preparada por Chiharu Sano[4]. A vagueza da informação nesta

[4] Consulta realizada em 12.11.99, no endereço ftp://ftp.ics.uci.edu/pub/machine-learning-databases/credit-screening/credit.lisp)

página da Web nos levou inclusive a montar a nossa própria exemplificação da aplicação do ID3, no domínio do crédito, para efeito de deixar claras as diferenças em relação aos nossos próprios modelos baseados na similaridade cognitiva de casos de crédito.

7.6.4.1 "*Japonese credit screening*": Chiharu Sano

O algoritmo ID3 implementa a aprendizagem de máquina, construindo uma árvore de discriminação (cf. seção 2.6.2) e fazendo uso desta árvore para gerar regras generalizadoras a guiarem decisões de crédito. Chiharu Sano, na página citada, apenas *codificadamente* apresenta: *(i)* os exemplos positivos e negativos de crédito que foram considerados (créditos concedidos e não concedidos); *(ii)* exemplos de regras geradas pelo sistema; e *(iii)* os atributos para cada exemplo de crédito (dez atributos ao todo: finalidade, emprego, área residencial, idade, estado civil, sexo, dinheiro em depósito, amortização, tempo de amortização, permanência no emprego).

Um exemplo dessas regras para negar crédito – aqui expressa em Lisp – é a seguinte regra, quase discriminatória em relação à mulher, mas que não deixa de ser representativa da cultura japonesa:

```
(def-rule jobless_unmarried_fem_reject
 (((jobless_unmarried_fem_reject ?s)
   (jobless ?s)
   (female ?s)
   (unmarried ?s))))
```

Ou seja: um empréstimo deve ser negado, sempre que o seu solicitante estiver desempregado, se esse solicitante for do sexo feminino e se estiver não casada. A pesquisa japonesa, no entanto, não descreve a construção da respectiva árvore de discriminação capaz de dar origem a regras como esta ou capaz de ensinar uma *lição de crédito*, como veremos na seqüência.

7.6.4.2 Exemplos positivos e negativos de treinamento

Mesmo o pesquisador Chiharu Sano não analisando a sua aplicação de ID3 ao crédito, subentende-se que ele tenha partido de exemplos positivos e negativos de crédito – na terminologia da *aprendizagem indutiva*. Exemplos positivos e negativos de crédito em indução correspondem – na terminologia do RBC – por exemplo, a casos de créditos aprovados ou negados e de casos de créditos com prejuízos. Podem corresponder também a casos de alta qualidade e de baixa qualidade decisória, tal como em nossos experimentos.

Admitamos, para efeito de nossa exemplificação do ID3 aplicado ao crédito, os seguintes exemplos positivos e negativos de crédito (ou casos de lucro ou prejuízo no empréstimo devido à qualidade da decisão tomada):

	Resultado	*Amortização*	*Emprego*	*Rendimento*
Caso 1	Lucrativo	R$200	Assalariado	R$2000
Caso 2	Muito Prejuízo	R$600	Assalariado	R$4000
Caso 3	Muito Lucrativo	R$300	Horista	R$3000
Caso 4	Prejuízo	R$400	Assalariado	R$1500

Figura 7.11: Casos positivos e negativos de crédito

Dois destes casos da Figura 7.11 são resultantes de decisões de *boa qualidade* e renderam lucros ao emprestador (*Caso* 1 e *Caso* 3); enquanto outros dois são resultados de decisões de má qualidade e ocasionaram prejuizos ao emprestador (*Caso* 2 e *Caso* 4). ID3 atua sobre os atributos destes empréstimos através da:

- construção da árvore de discriminação, a partir dos exemplos dados; e do
- emprego desta árvore de discriminação *(i)* ora para gerar regras; *(ii)* ora para ensinar lições.

7.6.4.3 Que empréstimos passados apoiam um empréstimo novo?

A construção da árvore de discriminação consiste em:

(i) Descrever todos os exemplos ou casos em termos dos seus atributos;

(ii) Repetir até que todos os casos estejam divididos em atributos simples...

(ii.1) Tomar um atributo;

(ii.2) Dividir os casos, segundo a presença ou ausência deste atributo tomado.

Ou seja: o algoritmo tem de empregar uma heurística para selecionar aquele atributo mais promissor em cada momento. Esta heurística é chamada de *ganho de informação*, e está baseada na função de entropia de Shannon. Em cada nó da árvore de discriminação criada, ID3 é capaz de encontrar entre os atributos dos casos dados, aquele atributo que possua a propriedade de melhor dividir ou discriminar, um do outro, os casos da base mostrada na Figura 7.11. Admitamos que essa medida de *ganho de informação* venha a estimar o atributo *Resultado* como sendo o atributo alvo do processo, uma vez que dele deve se esperar lições importantes sobre decisões tomadas no passado. Nos interessa encontrar aquele particular atributo dos casos que seja capaz de responder: *Como foi o empréstimo para o emprestador*? Ou, perguntando de outro modo: *Que empréstimos tiveram um resultado lucrativo para o emprestador*? Será a resposta a esta indagação que terá a propriedade de guiar uma nova decisão de qualidade sobre um novo empréstimo, na perspectiva lógica do RBC.

Primeiro nó da árvore de discriminação

Como o atributo *Resultado* (o atributo alvo) vai ser influenciado pelos demais atributos e valores (pois as variáveis em crédito não são ao todo independentes) será preciso então encontrar uma característica ou atributo que, ao dividir ao meio o conjunto de casos, também seja capaz de predizer sobre o atributo *Resultado*. O atributo *Rendimento* (na Figura 7.11) não será capaz de nos predizer um resultado favorável, pois, tanto o maior rendimento (4.000) quanto o menor rendimento (1.500) levam a casos contendo histórias de prejuízo. O atributo *Emprego*, por sua vez, também não prediz uma boa decisão: pois o mesmo atributo leva tanto a casos com história de resultado favorável quanto a história de casos com resultado desfavorável de empréstimo. O atributo *Amortização* será o melhor discriminador. Todas as amortizações iguais ou maiores de 400 justificam ou levam a casos de prejuízo para o emprestador. Chegamos então a uma primeira divisão daqueles casos da base (criando o primeiro nó na árvore de discriminação) conforme mostra a Figura 7.12.

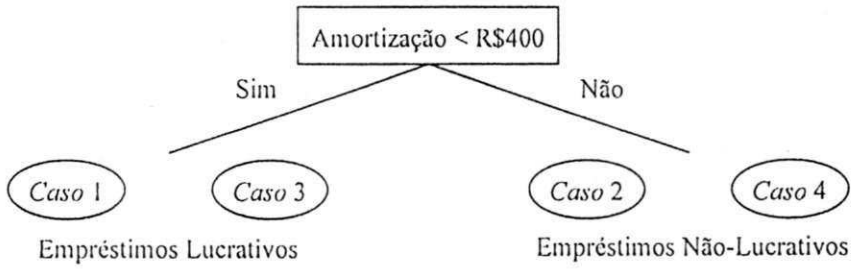

Figura 7.12: Primeiro nó na árvore de discriminação

Demais nós da árvore de discriminação

Partindo deste primeiro nó, será preciso agora discriminar aqueles empréstimos cujas decisões de concessão levaram a bons resultados (discriminar entre empréstimo bom e muito bom ou entre *Caso* 1 e *Caso* 3). Esses empréstimos (*Caso* 1 e *Caso* 3) apresentam os atributos *amortização* e *rendimentos* similares, porém, o atributo *emprego* do cliente tomador é um bom discriminador entre empréstimo *bom* e empréstimo *muito bom*. Fazendo-se deste atributo *emprego* (que é bom discriminador) o segundo nó que se procurava, tem-se agora a nova configuração desta árvore mostrada na figura 7.13.

Também os casos de empréstimos resultantes de decisões de má qualidade (caso 2 e caso 4) precisam ser disciminados em empréstimo causadores de *muito prejuízo* e simplesmente empréstimo com *prejuízo*. Aqui, os clientes apresentam a mesma situação de *Emprego* e uma alta *Amortização*. Discriminando-os pelo atributo *Rendimento* (o melhor da situação) constroi-se então uma árvore de discriminação completa para os casos dados, como mostra a Figura 7.14.

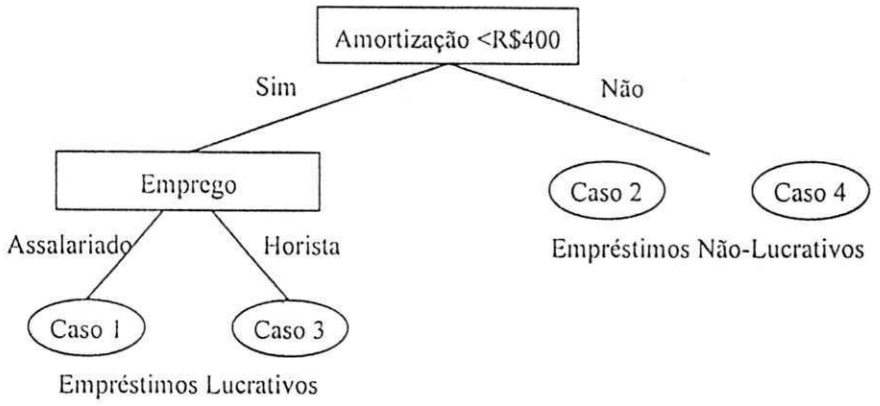

Figura 7.13: Segundo nó na árvore de discriminação

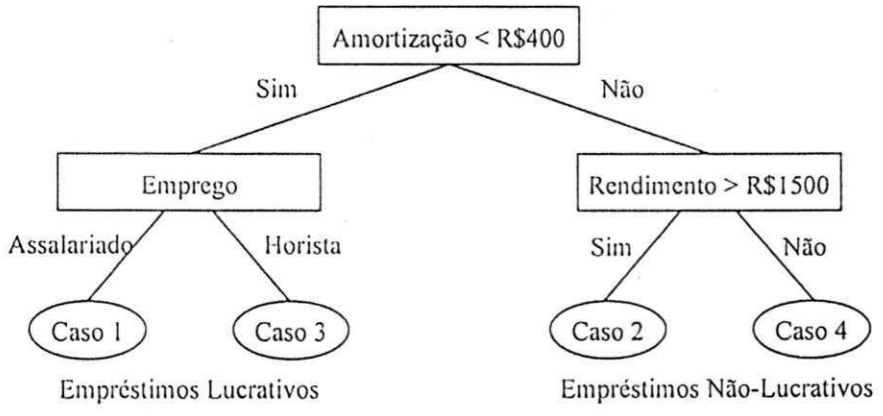

Figura 7.14: Árvore de discriminação completa

7.6.4.4 Uso da árvore de discriminação

O próximo passo do algoritmo será o emprego da árvore acima construída. Suponha que o analista de crédito esteja à frente de um novo cliente potencial – o *Caso* X – que apresenta as pré-condições para um empréstimo planejado tal como mostradas, na Figura 7.15.

	Resultado	*Amortização*	*Emprego*	*Rendimento*
Caso X	?	R$250	Assalariado	R$2500

Figura 7.15: *Caso X* de crédito a ser julgado segundo ID3

A árvore da Figura 7.14 então vai servir para nela se encontrar um caso SIMILAR QUANTO AO RESULTADO e que possa se acoplar (no todo ou em parte) ao *Caso* X do analista, referente a este cliente novo em apreço. A árvore é empregada de um modo puramente indutivo se considerarmos que os nós da árvore correspondem a *indagações* feitas durante uma consulta e que as folhas correspondem a *respostas* a estas indagações. Percorre-se essa árvore indagando em cada nó, confor-

me o algoritmo citado na seção 2.7.6. Indaga-se a partir do nó raiz da árvore (Faz-se: N = Raiz), concretamente:

1: A *Amortização* planejada/calculada para o novo cliente é menor do que 400? Como a resposta vai ser positiva, segue-se indagando no nó do ramo esquerdo da árvore.

2: *Assalariado* ou *horista* corresponde ao vínculo empregatício do novo cliente X? Como a Figura 7.15 diz tratar-se de um candidato(a) assalariado(a), chega-se então ao *Caso* 1 constante da árvore, o qual representa o melhor acoplamento para aquele novo caso de empréstimo da Figura 7.15 e para o qual se procurava um similar na árvore de discriminação (ou de decisão).

O *Caso 1* da árvore ensina ao tomador de decisão que esse empréstimo concedido no passado – nas condições em que foi negociado – rendeu lucros ao agente emprestador.

7.6.4.5 Síntese analítica da aplicação de ID3

O *Caso* X a ser decidido pelo analista será então um caso mais agrupável ao *Caso* 1 armazenado em árvore (visto que se acoplaram ao percorrer-se a árvore de decisão). O valor do atributo *Resultado* no *Caso* 1 da árvore pode ser transferido como lição para o *Caso* X (pode ser apropriado pelo analista para uso nesse novo caso) de tal modo a poder explicar o novo empréstimo como sendo um empréstimo que dará lucro ao emprestador. Portanto, esta lição encontrada na base de casos (no *Caso* 1) tornar-se-á uma boa razão para a aprovação do novo empréstimo por parte do agente financeiro, como mostra a Figura 7.16.

	Resultado	*Amortização*	*Emprego*	*Rendimento*
Caso X	Lucrativo	R$250	Assalariado	R$2500

Figura 7.16: *Caso Y* de crédito a ser decidido como *lucrativo*

Tome-se agora outra situação planejada de empréstimo e que está representada no *Caso* Y, da Figura 7.17 (representada no *Exemplo* Y, na terminologia da comunidade de aprendizagem indutiva). O que foi verdadeiro para o *Caso* X acima não se verificará em relação ao *Caso* Y. De fato, ao percorrer-se a mesma árvore, seguindo o mesmo procedimento anterior, chega-se ao *Caso* 2 da árvore, contendo a lição de que o empréstimo representado neste *Caso 2* redundou em um empréstimo com a característica de *muito prejuízo*.

	Resultado	*Amortização*	*Emprego*	*Rendimento*
Caso Y	?	R$3500	Assalariado	R$500

Figura 7.17: *Caso* de crédito a ser decidido sob risco de *grande prejuízo*

A aplicação do algoritmo ID3 vai então levar à conclusão de que – por agrupamentos/*clusters* de exemplos e por discriminação de atributos foi possível chegar-se ao *Caso* 2 e ao seu atributo *Resultado*. Mediante esse percorrimento na árvore, a nova solicitação de empréstimo representada no *Caso* Y vai ser decidida negativamente, portanto, decidida como sendo uma solicitação a dar prejuízo ao emprestador.

7.6.4.6 Indução em crédito × casos de crédito

O algoritmo indutivo acima descrito apresenta vantagens e desvantagens:

- É um algoritmo simples e também é um algoritmo rápido, independentemente de domínios onde venha a ser aplicado. Em nossa metodologia baseada na similaridade de casos de crédito também o tempo de processamento não constitui maiores problemas muito embora este tempo não tenha sido, cronometricamente, avaliado.

- Em ID3 um atributo deve ser usado para selecionar subconjuntos de exemplos (casos) que são então linearmente escaneados. Porém, não existirá nenhuma possibilidade de incorporar algum método de emprego daquele atributo ao próprio mecanismo interno de busca do ID3. Por exemplo, numa árvore de atributos (a árvore do ID3) não será possível re-visitar um certo nó [ALT 95, p. 8]. Em nossas metodologias, ao contrário, o modelo IBT (que lida com atributos) foi concebido para, mutuamente, se acoplar ao mecanismo de similaridade $SIM(m,p)$.

- Em ID3, a representação de conhecimento que é boa para computadores, seguramente, não é boa para possível inspecção a ser efetuada pelas pessoas (pelo analista de crédito, particularmente). Em nossas metodologias, contrariamente, os casos de crédito representam conhecimentos de um modo que facilita a compreensão humana.

- Em ID3 o conhecimento sobre o crédito fica pulverizado por toda a árvore. Pior ainda: informações relevantes para uma classificação ficam subordinadas a condições por vezes irrelevantes. Em casos de crédito, contrariamente, um caso de crédito funciona como um *cache* semântico – um compacto objeto semanticamente significativo.

- Várias árvores podem ser geradas a partir de um mesmo conjunto de exemplos de crédito; a ordem de apresentação dos atributos deve ser considerada importante, sendo uma regra geral apresentar, inicialmente, aqueles atributos mais importantes. Em nossos experimentos, contrariamente, a ordem de inserção dos atributos não introduz qualquer perturbação à modelagem e à manipulação dos casos.

7.7 Conclusão

Ao investigarmos acima as metodologias de indução, de *credit scoring*, redes neuronais e sistemas baseados em regras, objetivou-se um duplo resultado: *(i)* mostrar como esta problemática do crédito tem desafiado diferentes paradigmas da computação inteligente; e *(ii)* mostrar a diferenciação de abordagem propiciada pela aplicação de nossos modelos ao mesmo problema.

Mostramos como o domínio da *análise empírica* do crédito se revela um domínio apropriado – como bancada de teste (*testbed*) para os nossos modelos de indexação (de atributos, valores e pontagem por votos), de representação e de busca de similaridade de casos do paradigma de RBC. A tarefa inicial consistiu em um esforço de compreensão da Análise de Crédito e na concepção e operacionalização de um modelo de crédito não computacional que pudesse ser acoplado aos nossos formalismos de casos. Esta modelagem não computacional da decisão sobre crédito se tornou necessária dada a característica da Análise de Crédito de ser, por natureza, um domínio de fraca teorização.

Conclusão e trabalhos vindouros

Considerações finais

Toda a investigação aqui descrita objetivou cobrir problemas básicos da tecnologia de RBC com ênfase para o problema da busca de similaridade de casos, com fundamento em princípios ou teorias. Os trabalhos realizados, em síntese, compreendem:

- Um levantamento do estado da arte do RBC, destacando problemas ainda em aberto e destacando, nesse estado da arte, a computação da similaridade de casos;
- O desenvolvimento de metodologias de RBC apoiadas na importação do modelo teórico de similaridade de Tversky-Gati;
- O desenvolvimento de experimentos para validar, em domínio de verdade, os aspectos centrais dos conceitos a embasarem os modelos propostos.

A investigação vai além da propositura de novas metodologias de computação de casos. A postura analítica adotada durante a investigação contribuiu também para clarificar a base conceitual dessa tecnologia emergente e contribuiu para a solidificação de quatro conclusões principais que são:

(i) Os conceitos basilares dessa nova tecnologia de casos estão ainda em plena formação. Isto em parte explica a enorme disparidade na interpretação de conceitos, termos e até mesmo de processos do RBC.

(ii) A similaridade e a sua mensuração são fenômenos que acarretam os maiores impactos sobre os demais processos do RBC. Mostramos como a indexação de casos e demais processos são dependentes desta mensuração, o que, de fato, teoricamente já era admitido desde o início.

(iii) A importação para o RBC de um modelo teórico psicológico tal como o modelo de Tversky-Gati foi claramente viabilizado, como ficou demonstrado pela operacionalização computacional deste modelo e de suas extensões.

(iv) A computação da similaridade de casos, de fato, envolve um duplo papel dentro do

RBC, o que ainda não está muito claro para muitos dos desenvolvedores. A busca da similaridade tanto pode ser processada em tempo de resgate quanto em tempo de seleção de casos.

Deixamos porém este último aspecto da similaridade de casos como uma proposta de pesquisa futura enunciada na seção que segue sobre a propositura de novas linhas de investigação.

Trabalhos futuros

Aspectos importantes para a similaridade e a Computação Baseada em Casos ficaram ainda fora do escopo da presente investigação. Enunciamos alguns desses aspectos por nós não examinados, na forma de propostas de novas linhas de investigação nascentes dos trabalhos aqui realizados. Estas linhas de trabalho são:

Proposta 1: Similaridade para seleção de casos (pós-resgate)

Entende-se a seleção de casos como uma operação decorrente da operação de resgate de casos. Ao serem resgatados, os casos podem ser numerosos. Conseqüentemente, uma avaliação adicional se fará necessária como base para uma certa solução. A proposta, basicamente, consiste em examinar os efeitos da aplicação da métrica $SIM(m,p)$ em tempo de seleção de casos e não em tempo de resgate, a partir da memória (como o fizemos).

Proposta 2: Comparação de $SIM(m,p)$ com outras métricas comparáveis

Na investigação realizada, a preocupação maior consistiu em comparar o conjunto dos modelos propostos com outros paradigmas da computação inteligente aplicados ao domínio de experimentação. A proposta agora consiste em: *(i)* identificar na literatura outras abordagens cognitivas de similaridade quer na área de RBC quanto fora dela (em Analogia, por exemplo); *(ii)* identificar termos de comparação entre as métricas possivelmente a encontrar; e *(iii)* experimentar com essas métricas, comparativamente.

Proposta 3: Resgate paralelo usando $SIM(m,p)$

Na experimentação realizada, a aplicação de $SIM(m,p)$ para efeito de resgate de casos deu-se seqüencialmente. Porém, nada impede que $SIM(m,p)$ possa ser aplicada simultaneamente sobre todos os casos que estejam na memória.

Proposta 4: IBT na indexação de bases de casos navegáveis

As bases de casos navegáveis são bases de conhecimento em algum domínio de interesse que dispensam manipulações do tipo resgate, procura por similaridade, adapta-

ções, etc. Elas apenas admitem serem inspecionadas e, nesse aspecto, são de grande valia.

Proposta 5: Ontologias para a Análise de Crédito

A Engenharia de Ontologia, como já estão sendo chamados os estudos sobre a formalização de terminologias e conceitos, é ramo emergente importante da Engenharia de Conhecimento. A proposta consiste em ensaiar com a construção de ontologias aplicáveis ao domínio da análise empírica das operações de crédito empregando para isto ferramentas apropriadas tal como a lógica das descrições exemplificada nesta investigação.

Proposta 6: Aplicação no domínio de avaliação da qualidade de vida e de projetos comunitários

Apontamos, na investigação, este domínio como sendo um candidato também apropriado à aplicação das metodologias desenvolvidas. Novas experimentações comprovariam ou não esta viabilidade.

Proposta 7: Flexibilização da interface com a base de casos

Nosso interesse na experimentação, de imediato, se voltou para a obtenção dos resultados sobre a viabilização de *IBT*, *SIM*(m,p), e *ORDEN*. Porém, ensaios interessantes podem ser feitos para permitir ao usuário maior flexibilidade ao explorar a base de casos. Permitir, por exemplo, discriminar casos pela apresentação de atributos adicionais aos inicialmente apresentados; rever um caso apresentado como *input* inicial; fazer um *zoom* para um certo caso; permitir grupar os casos resgatados segundo sejam os primeiros melhores, os segundos melhores, etc.

Proposta 8: Implementação do modelo *AVAL*

A avaliação de bases de casos requer que esta mesma base já tenha uma vida útil considerável, uma quantidade de casos razoável e também o depoimento de um bom número de usuários. Enfim, a avaliação através de métricas requer um *log* das interações com o sistema. Estabelecidos estes requisitos, uma proposição a ser desenvolvida será a implementação do modelo de métricas para a avaliação da qualidade dos casos e da base de casos criada a partir dos casos-semente fornecidos nesta investigação.

Conclusão

Foram mostradas na Introdução geral deste documento as principais contribuições alcançadas por esta investigação, em termos de contribuições para a concepção de similaridade cognitiva de casos e em termos das contribuições metodológicas desenvolvidas. Acrescentamos, aqui, as conclusões

mais gerais de nossa investigação, como também as possíveis extensões dela decorrentes em termos de propostas para futuros trabalhos.

Bibliografia

[AAM 91] Aamodt, A. *A Knowledge-Intensive Approach to Problem Solving and Sustained Learning*. University of Trondheim, Norway, 1991.

[AAM 96] Aamodt, A.; Plaza, E. "Case-Based Reasoning: Foundational Issues, Methodological Variations, and System Approaches". http://www.iiia.csic.es/People/enric/AICom.html#RTFT.C1.

[ALB 00] Albuquerque, A. R. R. *Proposta de um Modelo de Automação do Planejamento da Qualidade Através da Utilização de Sistemas de Help Desk que Empregam Raciocínio Baseado em Casos*, Dissertação de Mestrado em Informática, UFPB, Campina Grande (PB), 2000. (*a aparecer*)

[ALT 95] Althoff, K.; Eric, A.; Ralph, B.; Michael, M. *A Review of Industrial Case-Based Reasoning Tools. An AI Perspectives Report*. AI Perspectives, Goodall A. (Ed.), Oxford (RU), 1995.

[AND 97] Andreasen, T.; Christiansen, H.; Larsen, H. L. "Introduction". In: *Flexible Query Answering Systems"*. Andreasen, T.; Christiansen, H.; Larsen, H. L. (Eds.). Kluwer Academic Publishers, Londres (RU), pp. ix-xiv, 1997.

[BAA 99] Baader, F.; Molitor, R. "Rewriting in Description Logics Using Terminologies". *Proceedings of the International Workshop on Description Logics (DL'99)*, pp. 76-80, Linköping, Sweden, julho 1999. http://SunSITE.Informatik.RWTH-Aachen.DE/Publications/CEUR-WS/Vol-22/

[BAR 89] Bareiss, R. et al. Discussion on "Similarity Metrics". In: *Proceedings of the Case-Based Reasoning Workshop*, pp. 67-71, Pensacola Beach, Florida (EUA), Morgan Kaufmann, 1989.

[BIT 98] Bittencourt, G. *Inteligência Artificial: Ferramentas e Teorias*. Editora da UFSC, Florianópolis (SC), 1998.

[BRO 95] Brown, M.; Filer, N. "Beauty vs. The Beast: The Case Against Massively Parallel Retrieval". Progress in CBR. In: *Lecture Notes in AI*, 1020. First UK Workshop in CBR, pp. 42-58, Proceedings, Salford, UK, 1995.

[BUA 97] Buarque, C. "O Ordenamento de Projetos Através de Pontagem". In: *Avaliação Econômica de Projetos*, pp. 231-242, Editora Campus, Rio de Janeiro (RJ), 1997.

[CHA 87] Charniak, E.; McDermott, D. *Introduction to Artificial Intelligence*. Addison-Wesley, Reading (EUA), 1987.

[CHE 87] Cheong, F.-C. *Internet Agents – Spiders, Wanderers, Brokers and Bots*. New Ridres

Publishing, 1996.

[COO 91] Cook, D. J. "The Base Selection Task in Analogical Planning". In: *Proceedings of the 12th. International Joint Conference on AI*, Vol. 2, pp. 790-795, Morgan Kaufmann, Sydney, Australia, 1991.

[FIR 95] Firt, C. et al. "A Framework for Analysis of Data Quality Research". *IEEE Transactions on Knowledge and Data Engineering*, 7(4):623-639, agosto 1995.

[FOR 86] Forsyth, R.; Rada, R. *Machine Learning: Applications in Expert Systems and Information Retrieval*. Ellis Horwood Limited, Chichester (RU), 1986.

[FOX 93] Fox, E. A. *Sourcebook on Digital Libraries*. Report for the National Science Foundation. Blacksburg (EUA), 1993, http://fox.cs.vt.edu/DLSB.html.

[FRA 99] Franconi, E.; Michael, K. (Eds.). *Proceedings of the 6th International Workshop on Knowledge Representation Meets Databases* (KRDB'99). Linköping (Suécia), julho 1999.

[GAT 99] Gates, B. *A Empresa na Velocidade do Pensamento: Com um Sistema Nervoso Central*. Companhia das Letras, São Paulo, 1999.

[GOD 99] Godfrey, Parke; Gryz, J. "Answering Queries by Semantic Caches". LNCS, 1677, (Ed): Trevor Bench-Capon, Giovanni Soda, A Min Tjoa; Database and Expert Systems Applications, 10th International Conference, (DEXA'99), pp. 485-497, Florence, (Italia), agosto/setembro 1999.

[GOM 98] Gomes, J. O. *Sistemas Help-Desk: Metodologias, Aplicações e Estudo de Caso*. Dissertação de Mestrado em Informática, UFPB, Campina Grande (PB), dezembro 1998.

[GOR 99] Gorgônio, F. Luz e *Uma Arquitetura de Sistemas Inteligentes de Apoio ao Usuário*, Dissertação de Mestrado em Informática, UFPB, Campina Grande (PB), julho 1999.

[HAM 89] Hammond, K. J. *Case-Based Planning: Viewing Planning as a Memory Task*. Academic Press, Boston, 1989.

[HDI 97] Help Desk Institute. *Help Desk and Customer Support Practices Report*. Colorado Springes (EUA), 1997.

[IBC ??] Instituto Brasileiro de Ciência Bancária. *Manual de Política e Processo Decisório de Crédito*. 2ª Edição, Cadernos IBCB 13, São Paulo, SP.

[KOL 91] Kolodner, J.; Menachem, Y.J. *Case-Based Reasoning: An Overview*. The Institute for the Learning Sciences, Northwestern University, Evanston (EUA), 1991.

[KOL 93] Kolodner, J. *Case-Based Reasoning*. Morgan Kaufmann Publishers, Inc., 1993.

[KOL 96] Kolodner, J. "Making the Implicit Explicit: Clarifying the Principles of Case-Based Reasoning". In: **[LEA 96]**, pp. 349-370.

[KOW 97] Kowalski, G. *Information Retrieval Systems: Theory and Implementation*. Kluwer Academic Publishers, USA, 1997.

[KUM 92] Kumar, Subrata. *Deductive Databases and Logic Programming*. Addison-Wesley

Publishing Company, Cambridge , (UK), 1992.

[LEA 96] Leake, D. B (Ed.). *Case-Based reasoning: Experiences, Lessons, and Future Directions*. The MIT Press, (EUA), 1996.

[LEW 94] Lewis, E.M. *Introduction to Credit Scoring*. Fair, Isaac and Co., Inc, (EUA), 1994.

[LUG 93] Luger, G.F.; Stubblefield, W. A . *Artificial Intelligence: Structures and Strategies for Complex Problem Solving*. Benjamin/Cummings, Redwood City (EUA), 1993.

[MAH 95] Maher, Mary Lou; Balachandran, M. Bala; Zhang, Dong Mei. *Case-Based Reasoning in Design*. Lawrence Erlbaum Associates, Inc., Publishers, N.J (EUA), 1995.

[MAK 96] Mark, W.; Simoudi, E.; Hinkle, D. "Case-Based Reasoning: Expectations and Results". In: [**LEA 96**], p. 281.

[MAR 96a] Martins, A.; Lula, B.; Ferneda, E. "Computing Analogies Through Logic". *Procee dings (Student Session) of the XIIIth Brazilian Symposium on Artificial Intelligence - SBIA '96*, pp. 33-38, Curitiba (PR), 1996.

[MAR 96b] Martins, A.; Ferneda, E. "Aprendizagem por analogia na máquina e nas pessoas", *Revista de Informática Teórica e Aplicada*, Vol. 3, nº 1, pp. 23-38, ISSN 0103-4308, Instituto de Informática - UFRGS, Porto Alegre (RS), 1996.

[MAR 97] Martins, A.; Ferneda, E. "Arquiteturas de apoio à explicação: o projeto LEGIS", *I Encontro Nacional de Inteligência Artificial – ENIA'97*, pp. 177-184, XVII Congresso Nacional da Sociedade Brasileira de Computação (SBC'99), Brasília (DF), agosto 1997.

[MAR 99a] Martins, A.; Ferneda, E. "Case-Based Query Answering: A Model Using Cognitive Similarity". In: *Proceedings of the 6th International Workshop on Knowledge Representation Meets Database (KRDB'99)*, pp. 36-40. [Affiliated Event with the 16th International Joint Conference on Artificial Intelligence (IJCAI'99)], Linköping (Suécia), julho 1999.

[MAR 99b] Martins, A.; Ferneda, E. "Computing Similarity Among Cases: An Operational Model Using Cognitive Theory". In: *Proceedings of the Argentine Symposium on Artificial Intelligence (ASAI'99)*, pp. 167-176, 28th Jornadas Argentinas de Informatica e Investigación Operativa, Buenos Aires (Argentina), setembro 1999.

[MAR 99c] Martins, A.; Ferneda, E. "The CBR Fundamental Problem: A Case Indexing Model Using *Credit Scoring*". *Anais do XIX Congresso Nacional da Sociedade Brasileira de Computação (SBC'99) – Vol. 4: II Encontro Nacional de Inteligência Artificial (II ENIA)*, pp. 369-381, Rio de Janeiro (RJ), julho 1999.

[MIN 89] Minsky, M. *A Sociedade da Mente*. Livraria Francisco Alves Editora S. A, Rio de Janeiro, (RJ), 1989.

[MON 95] 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, COPELE/UFPb, Campina Grande, (PB), 1995.

[MOT 97] Motro, A.; Rakov, I. "Not All Answers are Equally Good: Estimating the Quality of Database Answers". In: [**AND 97**], pp. 1-21.

[POR 89] Porto, A. "A Framework for Deducing Useful Answers to Queries". In: *Concepts and Characteristics of Knowledge-Based Systems.* Ed: M.Tokoro, Y. Anzai, and A. Yonezawa. Elsevier Science Publishers B. V., North-Holland, pp. 419-437, 1989.

[RAM 97] Ramalho, G. *Construction d'un Agent Rationnel Jouant du Jazz.* Tese de Doutorado, Université Paris VI, Paris (França), 1997.

[RIE 96] Riesbeck, C. K. "What Next? The Future of Case-Based Reasoning in Post-Modern AI". In: [LEA 96], pp. 371-388.

[RUS 95] Russell, S.; Norvig, P. *Artificial Intelligence. A Modern Approach.* Prentice-Hall, Inc., N.J, 1995.

[SAL 75] Salton, Gerard. *Dynamic Information and Library Processing.* Prentice-Hall, 1975.

[SLI 97] Sliwiany, R. M. *Sociometria.* Ed. Vozes, Petrópolis (RJ), 1997.

[SUG 98] Sugimoto, Masanori; et al. "A System for Constructing Private Digital Libraries Through Information Space Exploration". International Journal on Digital Libraries, Vol. 2, N. 1, October, p. 58, 1998.

[TAL 95] Talebzadeck, H. et al. "Countrywide Loan-Underwriting Expert System". *AI Magazine*, Volume 16, No. 1, Spring , pp. 51-64, 1995.

[TAN 91] Tanner, M. C.; Keuneke, A. M. "Explanations in Knowledge Systems: The Roles of the Task Structure and Domain Functional Models". *IEEE Expert*, pp. 50-57, Junho 1991.

[TVE 78] Tversky, A., Gati, I. "Studies of Similarity". In: *Cognition and Categorization.* Rosch, E., Lloyd, B. B. (Eds.), Lawrence Erlbaum Associates. Hillsdale (EUA), pp. 79-98, 1978.

[WAT 97] Watson, I. *Applying Cased-Based Reasoning: Techniques for Enterprise Systems.* Morgan Kaufmann Publishers, Inc. San Francisco (EUA), 1997.

[WIL 97] Williams, C. *Case Base Retrieval.* http://www.inference.com., 1997.

Anexo A

Métricas de similaridade global e local mais utilizáveis em RBC

Foram apresentados no Capítulo 2 (seção 2.7.9.2) conceitos básicos de similaridade de casos. Exemplificamos, neste Anexo A, as métricas de similaridade em RBC mais utilizáveis para se comparar casos computando as suas similaridades tanto locais quanto globais.

Medidas de similaridade local

Na tabela A.1, *a* e *b* estão designando conjuntos de valores assumidos por dois casos. A coluna *Valoração* indica se um certo atributo deve assumir um único valor ou se ele pode assumir uma lista como sendo uma lista de valores.

Medidas de similaridade global

Uma vez um conjunto de similaridades locais Sim_i tendo sido calculadas para cada atributo dos casos, será necessário combiná-las numa medida de similaridade global . As metricas apresentadas na Tabela A.2 sâo modelos de métricas globais ou agregadoras de similaridades locais.

Tabela A.1: Medidas de similaridade local

	Similaridade local	Tipo de atributo	Valoração
1.	$Sim(a,b) = \begin{cases} 0, \text{ se } a \cap b = \varnothing \\ 1, \text{ caso contrário} \end{cases}$	Simbólico	Monovalorado Multivalorado
2.	$Sim(a,b) = \frac{Card(a \cup b) - Card(a \cap b)}{Card(a \cup b)}$	Simbólico	Multivalorado
3.	$Sim(a,b) = \frac{Card(a \cup b) - Card(a \cap b)}{\min(a \cup b)}$	Simbólico	Multivalorado
4.	$Sim(a,b) = \frac{Card(a \cup b) - Card(a \cap b)}{\max(a \cup b)}$	Simbólico	Multivalorado
5.	$Sim(a,b) = \frac{Card(a \cup b) - Card(a \cap b)}{Card(O)}$	Simbólico	Multivalorado
6.	$Sim(a,b) = \frac{ec(\min(a^-,b^-),\max(a^-,b^-)) - Card(a \cap b)}{Card(O)}$	Simbólico ordenado e numérico	Multivalorado
7.	$Sim(a,b) = \frac{\lvert a-b \rvert}{ec(O)}$	Numérico	Monovalorado
8.	$Sim(a,b) = \frac{\lvert a_c - b_c \rvert}{ec(O)}$	Numérico	Multivalorado
9.	$Sim(a,b) = \frac{ec(\min(a^-,b^-),\max(a^-,b^-)) - ec(a \cap b)}{ec(O)}$	Numérico	Multivalorado
10.	$Sim(a,b) = \frac{ec(a \cup b) - ec(a \cap b)}{ec(a \cup b)}$	Numérico	Multivalorado
11.	$Sim(a,b) = \frac{ec(a \cup b) - ec(a \cap b)}{\min(ec(a),ec(b))}$	Numérico	Multivalorado
12.	$Sim(a,b) = \frac{ec(a \cup b) - ec(a \cap b)}{\max(ec(a),ec(b))}$	Numérico	Multivalorado
13.	$Sim(a,b) = \frac{2h(a \cup b) - h(b) - h(a))}{2h_{\max}}$	Simbólico e Numérico	Multivalorado
14.	$Sim(a,b) = \frac{h(\text{node merging } a \text{ and } b)}{\text{total height of tree } h}$	Simbólico e Numérico	Monovalorado

- Monovalorado: o atributo tem exatamente um valor por vez
- Multivalorado: o atributo toma uma lista como seus valores ou possivelmente um intervalo
- O Conjunto de possíveis valores para o atributo
- *Card* Tamanho de um conjunto
- a^-, b^- Limite superior de a (em relação a b)
- a_c Ponto central do intervalo a
- $ec(I)$ Valor absoluto entre limites superior e inferior do intervalo I
- h Profundidade numa hierarquia simbólica

Tabela A.2: Medidas de similaridade global

	Similaridade global	Nome
1.	$SIM(A,B) = \frac{1}{p}\sum_{i=1}^{p} Sim_i(a_i,b_i)$	Block-City
2.	$SIM(A,B) = \sum_{i=1}^{p} \omega_i Sim_i(a_i,b_i)$	Weighted Block-City
3.	$SIM(A,B) = \frac{1}{p}\sqrt{\sum_{i=1}^{p}[Sim_i(a_i,b_i)]^2}$	Euclidean
4.	$SIM(A,B) = \sqrt[r]{\frac{1}{p}\sum_{i=1}^{p}[Sim_i(a_i,b_i)]^r}$	Minkowsky
5.	$SIM(A,B) = \sqrt[r]{\sum_{i=1}^{p}\omega_i[Sim_i(a_i,b_i)]^r}$	Weighted Minkowsky
6.	$SIM(A,B) = \max_i \omega_i Sim_i(a_i,b_i)$	Maximum

Anexo B

"*Análise empírica*" de crédito em ciência bancária

B.1 Tomada de decisão em crédito

A tarefa fundamental da análise de crédito consiste em procurar definir o grau de risco envolvido em uma possível operação de crédito, sendo este grau de risco o fator decisivo para a concessão ou não de um determinado crédito financeiro. Os critérios considerados para as operações de crédito à empresas requerem o exame de inúmeros indicadores tais como:

(i) lucros do último semestre;

(ii) liquidez;

(iii) fluxo de caixa;

(iv) razão patrimônio/disponibilidades; e mais

(v) os chamados *Quatro C's* do crédito, etc.

Do ponto de vista de empréstimos a pessoas físicas, no entanto, costuma ser suficiente a análise daquilo que se costuma chamar os *Quatro C's* que compreendem [IBC ?]:

(i) *Caráter* e *Capacidade*: Variáveis Pessoais

(ii) *Capital* e *Condições*: Variáveis Financeiras

Estes *Quatro C's* são levados em conta tanto na análise de crédito realizada por especialistas humanos como também precisam ser consideradas em qualquer esforço de automação deste processo. Consideremos, portanto, estes fatores que, de uma forma ou de outra, estão presentes nos modelos computacionais de análise de crédito.

B.1.1 Caráter

Caráter é a determinação em honrar os compromissos assumidos. Mas, como medí-lo, para efeito de uma decisão envolvendo dinheiro? O caráter consiste na firmeza de vontade ligada à honestida-

de e que se reflete no esforço para cumprir uma obrigação. Um devedor pode chegar a se desfazer de bens essenciais para solver seus compromissos. Já outros não se dispõem a fazer qualquer esforço para tanto. É óbvio que os dois não possuem o mesmo caráter. Assim, o analista deve convencer-se de que o solicitante de crédito manterá a disposição de pagar e envidará todos os esforços para liquidar a dívida mesmo em condições adversas. Seja observado que não há taxa de juros ou bens vinculáveis em garantia que compensem o risco de se operar com uma pessoa que, reconhecidamente, não paga o que contrata. Sendo uma vontade, uma determinação, o caráter, no ponto de vista do crédito, não é absoluto. É mutável com o tempo e relativo quanto à situação ou aos valores envolvidos. Não estamos julgando somente se o cliente é ou não honesto, apesar de esse indicador ser um dos elementos para a decisão de concessão. Mesmo reconhecendo que o cliente possua idoneidade acima de qualquer suspeita, os valores colocados à sua disposição devem ser compatíveis com sua capacidade de pagamento. Independentemente da natureza do cliente (pessoa jurídica, produtor rural ou pessoa física) devem ser observados, entre outros, os seguintes aspectos:

(i) idoneidade, propriamente;

(ii) conceito de que desfruta;

(iii) pontualidade, e

(iv) alteração de comportamento ou de procedimento.

Informações sobre estes fatores, se restritivas, denotam que o proponente passou por alguma dificuldade que provocou os desabonos. Cabe ao emprestador examinar cada fato, seus reflexos e, principalmente, examinar qual tem sido a alteração de comportamento do cliente. A este respeito, compete ao agente financeiro trabalhar com a hipótese de que: nas horas de dificuldade é que melhor se demontra a personalidade.

B.1.2 Capacidade

É a habilidade, a competência empresarial ou profissional do proponente, bem como o seu potencial de produção e de aplicação do recurso tomado de empréstimo. Quando não houver convicção quanto à capacidade do proponente, a concessão do crédito estará configurando grande risco. Ainda que o cliente possua um caráter indiscutível e queira realmente honrar os compromissos, não terá como fazê-lo se não tiver capacidade. Na avaliação da capacidade do cliente devem ser observados, entre outros, os seguintes aspectos: formação profissional ou experiência na atividade, resultados alcançados em outras atividades (quando iniciante), habilidades administrativas, grau de tecnologia utilizada (no caso de empréstimo para empresas).

B.1.3 Capital

Admitindo-se que as *condições* sejam favoráveis, e que o cliente possui *capacidade* e seu *caráter* é indiscutível, deve-se ainda analisar o capital. Para o desempenho de qualquer atividade produtiva é necessário o emprego de recursos suficientes em instalações, máquinas, estoques, créditos, pessoal,etc. Esses recursos podem ser próprios (Patrimônio Líquido), de terceiros ou de ambos. O capital daquele proponente de uma operação de crédito deve ser compatível com a atividade desenvolvida e com o empréstimo proposto. No caso de o proponente de crédito ser uma pessoa física, o capital a ser avaliado inclui, entre outros, os seguintes aspectos:

- Rendimentos: salários, aposentadorias, pensões, receita líquida da atividade rural e outras fontes comprováveis;
- Composição das despesas: comprometimento da renda com aluguéis, prestação de casa própria, manutenção da família, etc.;
- Endividamento: montante, prazos e percentual do patrimônio comprometido com as dívidas.
- Evolução e qualidade do patrimônio: valor dos bens móveis e imóveis, aplicações financeiras, poupança, deduzido o endividamento;

B.1.4 Condições

Finalmente, as condições se referem ao microcenário/macrocenário em que o tomador de empréstimo está inserido. Levam em conta o momento atual em que o empréstimo é estudado e está para ser eventualmente desembolsado (tais como "pacotes econômicos", alteração no padrão monetário, abertura externa da economia, incentivos fiscais e reserva de mercado, controles sobre taxas de juros, câmbio, etc). *Condições* e *Capital* prestam-se a complementar os dois primeiros fatores. Idealmente, não se deve tomar decisões com base apenas em um dos 4 C's, isoladamente. Daí a necessidade de conceitos mais abrangentes capazes de englobar diferentes critérios da análise de crédito, tal como o conceito de risco, de limite de crédito, etc.

B.2 Metas da decisão sobre crédito

Do ponto de vista da tomada de decisões, uma análise a ser empreendida tem metas intermediárias antes da decisão final de conceder ou negar uma solicitação de crédito. Estas metas são o estabelecimento do risco do cliente, o enquadramento desse cliente na classe correspondente, o estabelecimento do limite do crédito e o estabelecimento da segmentação do limite de crédito. Na seqüência, detalhamos estes aspectos separadamente.

B.2.1 Risco de clientes de crédito

Risco é um elemento chave da análise de crédito. Concretamente, o risco é um conceito atribuído a determinado cliente a partir da comparação de suas características com determinados padrões de clientes bons e ruins. Em muitas agências financeiras adota-se a definição de risco como sendo uma *probabilidade de perda*. Os clientes então são classificados de acordo com esta probabilidade de perda que significa, para uma agência, os riscos calculados do cliente.

B.2.2 Classes de riscos

Quatro faixas de risco costumam ser estabelecidas para efeito de nelas posicionar um certo tomador: risco mínimo = *a*; aceitável = *b*; médio = *c*; considerável = *d*; alto = *e* (ver Figura 4.6). Quando dizemos que um cliente está na faixa de risco *a*, estamos dizendo que a probabilidade de esse cliente deixar de pagar um empréstimo é mínima. Já um cliente de risco *e* oferece uma probabilidade de perda tão alta que não compensa correr o risco de "bancar" o empréstimo. O problema do cálculo do risco será retomado mais à frente ao tratarmos da metodologia estatística mais comumente empregada pelas agências financeiras para a sua determinação.

B.3 Limite de crédito

A determinação do risco do cliente não objetiva tão somente enquadrar esse cliente em dada classe. A determinação do risco tem um papel prático na determinação do *Limite de Crédito* (LC) de um candidato a empréstimo. O LC é o valor máximo que uma agência emprestadora admite emprestar para determinado cliente. Em outras palavras, o LC é a exposição máxima ao risco do cliente admitida pelo emprestador. É importante notar que o LC não é tudo o que o cliente precisa nem tudo o que ele pode pagar. A atribuição do LC ao cliente tem por objetivo - além de definir a exposição do agente financeiro ao risco de um cliente - permitir uma *postura proativa*. O LC possibilita uma avaliação mais segura e, posteriormente, a agilidade no atendimento das propostas de empréstimos. Portanto, o LC se relaciona com o risco que indica a probabilidade de perda (mínima, média, considerável, etc) e é concedido em valor inversamente proporcional a esse risco oferecido pelo cliente. Quanto maior o risco do cliente, menor o limite e vice-versa, o que reduz a concentração do crédito em clientes de risco elevado.

B.4 Segmentação do limite de crédito

No caso de pessoas físicas, a atribuição de limites de crédito por parte de bancos, por exemplo, costuma ser segmentada por produto financeiro da instituição creditícia. Supondo-se que os pro-

dutos em apreço sejam, por exemplo, o cheque especial , o cartão de crédito e o crédito direto ao consumidor (CDC) um modelo de segmentação do crédito poderá ser:

- *Limite rotativo*: será definido em função do risco e da renda bruta mensal, englobando o limite para cheque especial e o limite do cartão de crédito. Modelos estatísticose de LC podem dar conta desta segmentação de tal modo a sugerirem, por exemplo, uma distribuição de 60% desse limite para o cheque especial e 40% para o cartão de crédito, que podem ser remanejados entre si.
- *CDC*: Crédito Direto ao Consumidor é o resultado de um percentual, definido em função do risco, aplicado sobre a renda bruta mensal . Refere-se ao valor máximo de prestação que pode ser comprometido mensalmente pelo cliente em operações de CDC.

Portanto, a análise de risco, o limite bem dimensionado e a adequada formalização dos créditos envolvem numerosas sub-tarefas. Enfrentá-las, em conjunto, é uma das condições indispensáveis para a manutenção da qualidade da carteira de crédito de uma instituição.

B.5 Paradoxo do cálculo de risco

Existem métodos de determinação e classificação de risco no mercado de dinheiro, tanto para o risco de pessoas jurídicas quanto para as pessoas físicas (por exemplo, o *Método de Monte Carlo* e o *Método de Preços-Sombra* [BUA 97]). Cada um deles oferece maior ou menor grau de confiabilidade, dependendo dos critérios utilizados e do rigor definido na classificação. É preciso observar nestes métodos algo interessante. Todo método de avaliação de risco tem um *paradoxo*: quanto mais rigoroso for este método, maior a quantidade de clientes bons que vão ficar fora da carteira de crédito de uma agência financeira; por outro lado, quanto mais flexível ele for, maior então será o número daqueles clientes ruins que serão atendidos com empréstimo. Observe-se a classificação de risco da seção B2.1(cf. Figura 4.6). Pode-se perceber que há uma situação onde clientes bons e ruins podem cair dentro de uma mesma faixa de risco (Faixa *c* da figura referida). Essa é a área da dúvida. No momento de classificar um cliente, não há uma clara definição se esse cliente tende a ser bom ou ruim. Como conseqüência deste processo de classificação, as políticas de crédito da empresa emprestadora têm de claramente estabelecer se ela deve correr mais riscos, *detendo maior participação no mercado*, ou menos risco, trabalhando somente com clientes comprovadamente bons. Optando por uma participação maior no mercado, um agente financeiro pode emprestar para todos os clientes bons e, nesse caso, também estará atendendo alguns clientes ruins que estão classificados na faixa *c*. Em condições muito especiais poderá trabalhar, também, com algum cliente*d*. Por outro lado, uma instituição financeira poderá ser conservadora, trabalhando somente com clientes reconhecidamente bons (faixas *a* e *b* da Figura 4.6). Neste caso, para garantir que não negoci-

ará com clientes ruins, deixará de atender, também, alguns bons que foram enquadrados na faixa *c*.

B.6 Metodologia de empréstimos: *credit scoring*

Em suma, todo emprestador de dinheiro quer se assegurar - por todo os meios possíveis - de que o seu dinheiro terá repagamento. O *credit scoring* - como uma metodologia estatística - tem sido um dos principais meios empregados na análise de concessão de créditos. Ela tem sido comumente usada para "quantificar" a contribuição de atributos de um algum objeto, cliente ou projeto econômico-financeiro quando estes atributos são dominantemente de natureza *qualitatitva* e de difícil mensuração quantitativa [BUA 97]. Ora, este é justamente o caso do crédito em que as agências de créditos tem de definir o risco de um cliente de crédito com base em variáveis ou atributos subjetivos e qualitativos [LEW 94]. O papel da metodologia de *credit scoring* é o de nortear a tomada de decisão sobre um crédito ou de servir de referência para os tomadores de decisão:

> *"Credit Scoring oferece somente uma indicação sobre o provável desempenho de um tomador de crédito – embora seja ela uma indicação muito boa. Mesmo sendo uma indicação, o credit scoring permite a uma organização creditícia acessar mais acuradamente se as amortizações definidas para um repagamento vão ser efetuadas, como combinado"*[1].

Portanto, a metodologia não garante a infabilidade de uma decisão a ser tomada sobre um cliente; mas, devidamente complementada pelo bom senso, o *credit scoring* tem o papel final de proteger os interesses tanto do emprestador como também do próprio tomador de empréstimos, na medida em que este tomador não queira entrar em dificuldades financeiras em decorrência de possiveis amortizações (repagamentos) que inapropriadamente tenham sido dimensionadas. Entre os aspectos importantes desta metodologia estatística se encontra aquele aspecto de como atribuir votos aos atributos dos objetos (dos empréstimos) em consideração. Uma vez que o *credit scoring* pode ser usado tanto para financiamentos por bancos e agências privadas quanto para os empréstimos a serem feitos com o dinheiro público, decorre daí que os atributos a receberem votos (pesos) durante o processo de avaliação vão variar dependendo da natureza do empréstimo a ser feito e da agência emprestadora. Dependendo da natureza do empréstimo, não somente vão variar *(i)* os atributos dos empréstimos, como também *(ii)* os seus valores, e *(iii)* os seus respectivos votos ou pesos. Para mostrar esta relação entre a metodologia de *credit scoring* e a *natureza* de possíveis decisões apresentamos, a seguir, o uso da metodologia em um empréstimo do tipo empréstimo público cujas características diferem bastante do crédito privado.

---

[1] *"Credit Scoring only gives an indication - although a very good one - of likely performance. However, it allows the Society to assess more accurately whether repayments will be met"*.

B.7 *Credit scoring* em empréstimos públicos

A Figura B.1 mostra um exemplo desta metodologia apresentada em documento da Sudene e que deveria ser aplicada por todos os avaliadores de empréstimo desta agência de desenvolvimento (*Critério de atribuição de pontos para determinar a prioridade de financiamento*). Aqui, o emprestador é o governo (representado pela Sudene, no exemplo) e o tomador deve ser julgado através do seu projeto de crédito. O *credit scoring* trata então de definição dos atributos (Atributos do Projeto), definição dos valores para estes atributos (Valores dos Atributos), atribuição dos pesos para estes valores (Pontuação) e, finalmente, do uso dos resultados da metodologia como mostrado a seguir.

Atributos do Projeto	Valores dos Atributos	Pontuação
Meta da produção	Energia	25
	Telecomunicações	25
	Bens de Capital	20
	Produtos Têxteis	10
	Agricultura	51
Localização	Estados mais Subdesenvolvidos	25
	Pernambuco e Bahia	15
Matérias-primas Locais	Mais de 80%	15
	Entre 50% e 80%	10
Tecnologia	Para aumentar produtividade	5

Figura B.1: *Credit Scoring* em empréstimos públicos

Nota-se, na Figura B.1, que a agência de financiamento utiliza o *credit scoring* para valorizar, sobretudo, os pedidos de finaciamento para a agricultura (51 votos), em estados mais subdesenvolvidos (25 votos) e cujos projetos empreguem uma maior quantidade de matérias-primas locais (15 votos). Para cada solicitação de financiamento obtem-se o N de votos que permitem a classificação do empréstimo nas categorias A, B, C, D e E. Ao permitir o estabelecimento destas categorias fica estabelecido também o volume a ser financiado como um percentual sobre o investimento total previsto no projeto. Tudo vai depender, portanto, da quantidade N de votos conseguidos pela proposta de financiamento. A Figura B.2 ilustra este resultado final da pontuação de atributos.

De acordo com este quadro, pode-se observar que todos os pedidos de crédito (projetos) que apresentem uma rentabilidade financeira razoável podem receber financiamento de no mínimo 30% dos investimentos totais. Mas, essa percentagem pode subir até 75% desses investimentos, sempre que a solicitação receba certa quantidade N de votos, determinados conforme a metodologia e os critérios estratégicos da agência.

Classes de Empréstimo	Pontuação	Financiamento (% sobre Investimentos)
A	$N \geq 50$	75
B	$40 \leq N < 50$	60
C	$30 \leq N < 40$	50
D	$25 \leq N < 30$	40
E	$N < 25$	30

Figura B.2: Relação entre classes de empréstimos, pontuação e financiamento

Estes são os aspectos fundamentais dos processos que compreendem a análise e concessão de empréstimos sobre os quais se apoia qualquer esforço de automação deste domínio.

Anexo C

Escala de valores para atributos de crédito

Os atributos das operações de empréstimos tomados em nossos experimentos assumem valores dentro de uma escala que varia de *A* a *D*, onde cada letra representa um grau ou uma caracterização particular de certo atributo. Esta escala de valores está estabelecida abaixo e se refere tanto a atributos para pessoas físicas tomadoras quanto para tomadores que sejam organismos e empresas.

Caráter & conceito do cliente

A – Cliente muito bem referenciado por todas as fontes (SERASA, SPC, etc) com clara indicação de que se trata de pessoa ou de organização digna de confiança (indicação positiva);

B – Cliente e empresas (e sócios) sem restrições na praça, mas sem maoires indicações positivas, ou seja, apenas não há indicações negativas;

C – Cliente, empresas (e sócios) com pequenas restrições cadastrais no passado, mas com situação já regularizada;

D – Clientes, empresas (e sócios) com históricos de restrições cadastrais e indicações negativas das partes consultadas.

Finalidade do crédito/financiamento

A – Empréstimo para pessoas físicas, sem finalidade especificada, com reembolso parcelado. *Hot money* ou empréstimo de curtíssimo prazo, para pessoas jurídicas já clientes do organismo emprestador;

B – Financiamento mediante projeto industrial. Alta qualidade da proposta, com sensível indicação de melhoria de competividade, produtividade e lucratividade;

C – Projeto, investimentos e prioridades de qualidade medíocre;

D – Investimentos de qualidade duvidosa, dependendo de hipóteses que não podem ser adequadamente avaliadas *a priori*.

Rendimento do cliente

A – Rendimentos acima de 15.000;

B – Rendimentos entre 15.000 – 10.000;

C – Rendimentos entre 10.000 – 5.000;

D – Rendimentos abaixo de 5.000;

Amortização mensal

A – No teto de 30% do rendimento;

B – Abaixo de 30% do rendimento do tomador;

C – Amortização baseada em dólar ou em taxas de juro de mercado;

D – Amortização com base na tabela *price*.

Capacidade

A – Tomadores (pessoas físicas) com formação e grande experiência na atividade que exerce;

B – Empresários e administradores com boa experiência administrativa;

C – Administradores iniciantes, devendo enfrentar dificuldades organizacionais;

D – Tomadores (pessoas ou empresas) sem experiência no ramo, sem estrutura administrativa.

Documentação

A – Documentação de alta qualidade. Empresas tomadoras com balanços e demonstrações contábeis fidedignas e coerentes.

B – Documentação relativamente adequada. Servindo basicamente para finalidades fiscais, e secundariamente para a administração da empresa;

C – Registros que não refletem integralmente a realidade. Corresponde a casos de empresas com contabilidade defasada.

D – Documentação confusa, incompleta e inconfiável.

Indicadores Financeiros

A – Ótimo índice de endividamento, de solvência, liquidez, rentabilidade, e outros que refletem adequadamente o passado e o presente do tomador de empréstimo;

B – Indicadores de bom nível, compatíveis com tomadores semelhantes;

C – Indicadores abaixo da média do setor ou de empresas semelhantes, mas ainda assim passíveis de melhoria;

D – Indicadores bastante baixos, indicando situação de dificuldades.

Garantias

A – Garantias de alta qualidade, tanto as reais quanto as fidejussórias; alta liquidez, nível baixo de depreciabilidade, elevado grau de controlabilidade e reduzidos custos de realização.

B – Garantias de boa qualidade, com o atendimento satisfatório dos parâmetros acima;

C – Garantias de qualidade razoável, prevendo-se dificuldades caso haja necessidade de execução dos créditos;

D – Garantias de baixa qualidade.

Nível de Risco da Operação

Níveis A, B, C, D, E e F, conforme descritos na Figura 7.2 do Capítulo 7.

Qualidade da Decisão de Concessão Tomada

A – Crédito "bem concedido". Inexistiu subseqüente deteriorização da situação financeira do cliente. Repagamento total conforme as cláusulas contratuais;

B – Crédito que forçou a necessidade de acompanhamento próximo, por descumprimento ou violação de cláusulas, ou pedidos de prorrogação;

C – Decisão de concessão foi posteriormente seguida de reexame e de uma nova decisão, a partir de deterioração da situação econômico-financeira do cliente detectada pelo acompanhamento. Houve esforços para recuperação do crédito;

D – Crédito "mal concedido". Esgotaram-se todos os meios de cobrança.

Anexo D

Sistema *SIM-Crédito* – uma sessão de busca de similaridade

No Capítulo 7 descreveu-se a aplicação dos modelos desenvolvidos à área do crédito financeiro ou empréstimo. O protótipo *SIM-Crédito* (um acrônimo para lembrar similaridades de casos de crédito financeiro) concretiza os resultados discutidos no capítulo referido.

De fato, não apenas a similaridade *SIM(m,p)* está sendo testada em *SIM-Crédito* mas todo aquele *framework* conceitual envolvendo a indexação necessária aos casos de crédito, envolvendo a similaridade cognitiva, propriamente e, envolvendo ainda, o *ranking* de casos modelado a partir de SIM(m,p). As telas mostradas nas figuras D.1 e D.2 (construídas em ambiente Java) ilustram uma sessão de busca de similaridade em *SIM-Crédito*. Através de *Opções 1* tem entrada no sistema o primeiro sub-conjunto de atributos de um caso de entrada (o caso-indagação). Correspondendo a cada atributo, tem-se a opção para a entrada dos valores de atributos e suas respectivas quantidades de votos. *Opções 2*, por sua vez, permite fazer a mesma coisa para o segundo sub-conjunto de atributos de um caso. Similarmente, o botão *Similaridade* permite a entrada dos parâmetros (a,b,c) vistos e a ativação do algoritmo de busca da similaridade também objeto de aplicação no Capítulo 7.

A tela mostrada na figura D.3 ilustra a apresentação para o usuário do ordenamento dos casos segundo as suas similaridades.

Qualquer caso de crédito já passado ao sistema pode ser revisitado quer para mera conferência de atributos, valores e votos quer para correções subseqüentes, como mostra a tela D.4, onde se vê os casos nomeados por números, porém, sem exibir as diagnosticidades dos atributos, que ficam reservadas ao projetista do sistema.

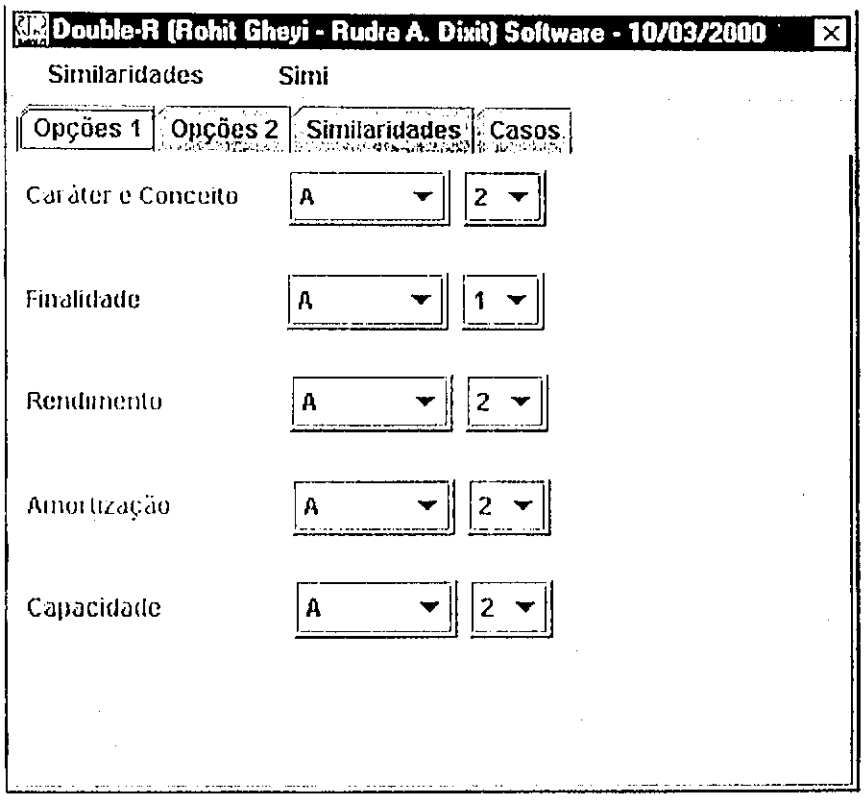

Figura D.1:Tela de entrada do caso-indagação (Opções 1)

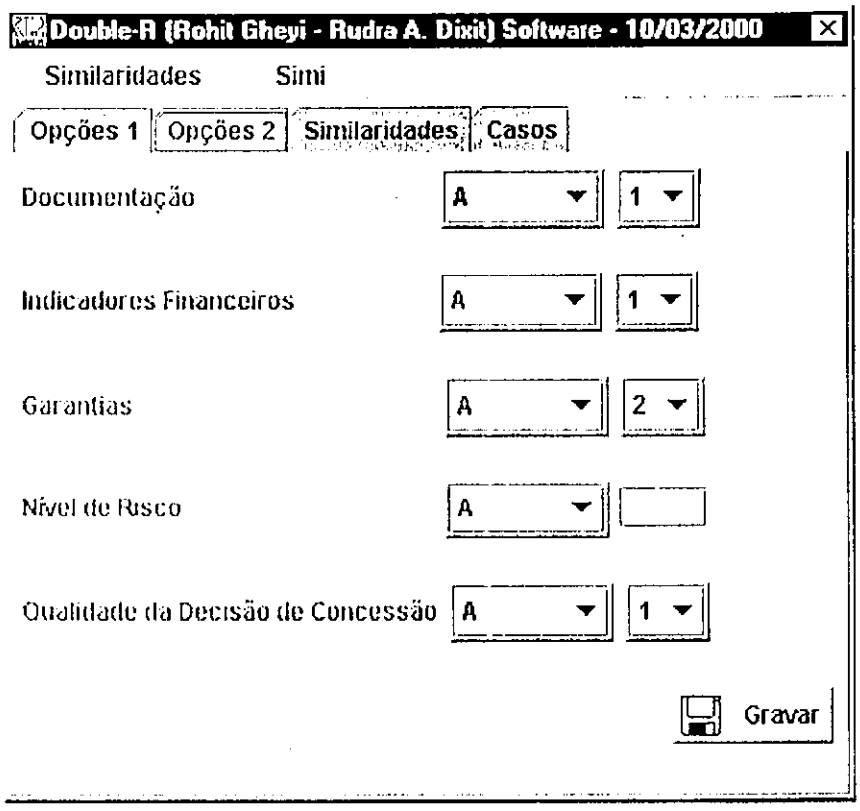

Figura D.2:Tela de entrada do caso-indagação (Opções 2)

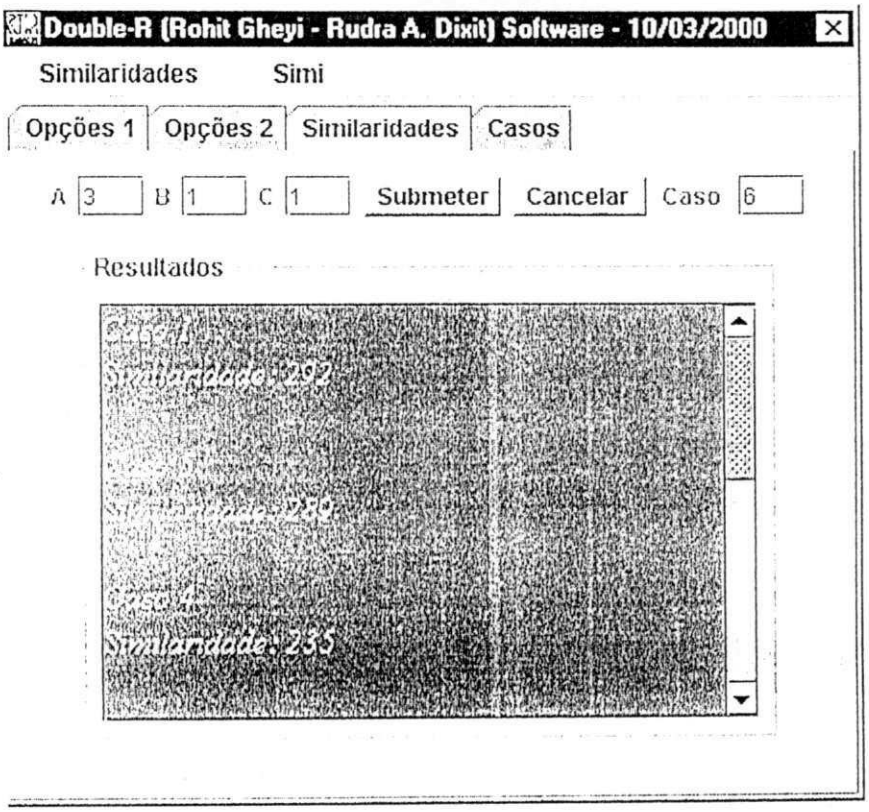

Figura D.3:Tela de resultados da busca e do *ranking*

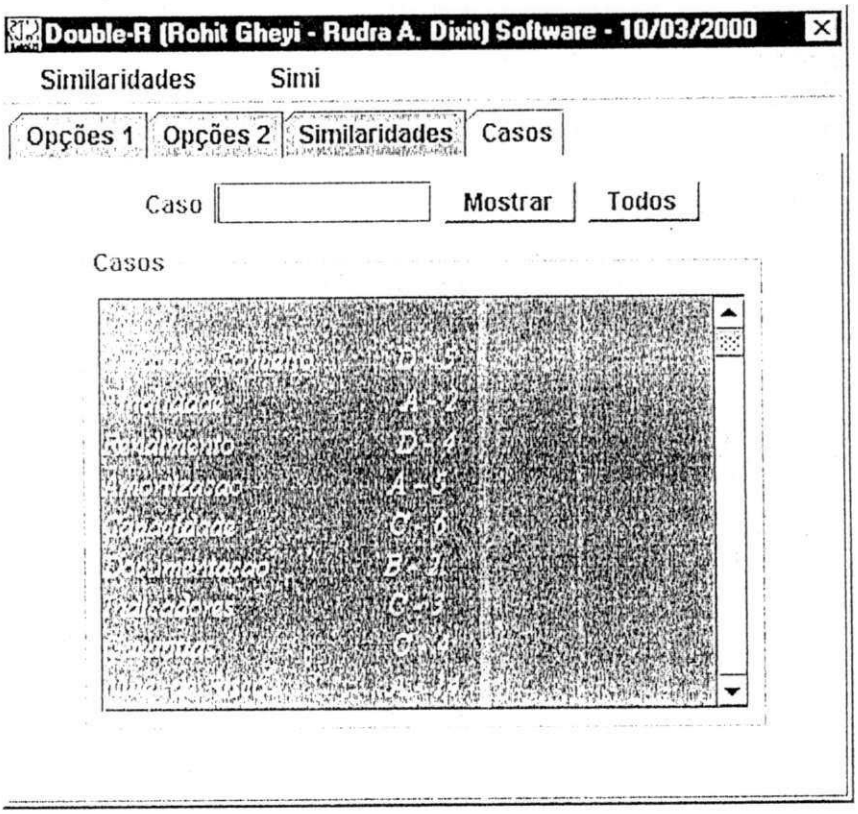

Figura D.4:Tela de exibição de casos

Anexo E

Casos-semente para experimentação

Casos-semente (*seed cases*) são aqueles a partir dos quais pode se expandir uma base de casos. Vinte (20) destes casos são apresentados, na seqüência, e estão a modelar alguma situação concreta de empréstimo financeiro – nosso domínio de experimentação.

Caso X1 /* Caso a ser decidido, na ausência apenas do atributo *Qualidade da Decisão* */		
Diagnosticidade-Atributo	**Atributo:Valor**	**Votos-valor-atributo**
	Concessão do Crédito	
2	Caráter: A	2
1	Finalidade: B	2
2	Rendimento: D	4
2	Amortização: C	6
2	Capacidade: A	2
1	Documentação: B	2
1	Indicadores: B	2
2	Garantias: D	6
3	Nível de risco: A	18
	Resultado da Concessão	
1	Qualidade da decisão: ?	?

Caso X2 /* Caso a ser decidido na ausência dos atributos *Capacidade* e *Qualidade da decisão* */		
Diagnosticidade-Atributo	**Atributo:Valor**	**Votos-valor-atributo**
	Concessão do Crédito	
2	Caráter: A	2
1	Finalidade: B	2
2	Rendimento: D	4
2	Amortização: C	6
2	Capacidade: ?	?
1	Documentação: B	2
1	Indicadores: B	2
2	Garantias: D	6
3	Nível de risco: A	18
	Resultado da Concessão	
1	Qualidade da decisão: ?	?

Caso 003 /* Caso de crédito com sucesso B onde houve boa qualidade de decisão */		
Diagnosticidade-Atributo	**Atributo:Valor**	**Votos-valor-atributo**
	Concessão do Crédito	
2	Caráter: D	5
1	Finalidade: B	2
2	Rendimento: D	4
2	Amortização: A	5
2	Capacidade: D	6
1	Documentação: B	2
1	Indicadores: C	3
2	Garantias: D	4
3	Nível de risco: A	14
	Resultado da Concessão	
1	Qualidade da decisão: B	2

Caso 004 /* Caso com insucesso D na decisão e *Nível de Risco* C */		
Diagnosticidade-Atributo	**Atributo:Valor**	**Votos-valor-atributo**
	Concessão do Crédito	
2	Caráter: D	5
1	Finalidade: A	2
2	Rendimento: D	4
2	Amortização: A	5
2	Capacidade: C	6
1	Documentação: B	2
1	Indicadores: C	3
2	Garantias: C	4
3	Nível de risco: C	28
	Resultado da Concessão	
1	Qualidade da decisão: D	3

Caso 005 /* Caso de crédito com insucesso D na decisão e ausência do atributo *Finalidade* */		
Diagnosticidade-Atributo	**Atributo:Valor**	**Votos-valor-atributo**
	Concessão do Crédito	
2	Caráter: D	5
1	Finalidade: ?	?
2	Rendimento: D	4
2	Amortização: A	5
2	Capacidade: C	6
1	Documentação: B	2
1	Indicadores: C	3
2	Garantias: C	4
3	Nível de risco: A	14
	Resultado da Concessão	
1	Qualidade da decisão: D	3

Caso 006 /* Caso de insucesso D e com "pontagem" fora do limite de aceitação */		
Diagnosticidade-Atributo	**Atributo:Valor**	**Votos-valor-atributo**
	Concessão do Crédito	
2	Caráter: D	8
1	Finalidade: A	4
2	Rendimento: D	8
2	Amortização: B	8
2	Capacidade: D	8
1	Documentação: B	4
1	Indicadores: C	4
2	Garantias: A	8
3	Nível de risco: C	35
	Resultado da Concessão	
1	Qualidade da decisão: D	4

Caso 007 /* Caso de crédito a funcionário público, com sucesso A de decisão */		
Diagnosticidade-Atributo	**Atributo:Valor**	**Votos-valor-atributo**
	Concessão do Crédito	
2	Caráter: A	2
1	Finalidade: A	1
2	Rendimento: D	4
2	Amortização: B	3
2	Capacidade: D	3
1	Documentação: B	2
1	Indicadores: D	4
2	Garantias: A	2
3	Nível de risco: A	14
	Resultado da Concessão	
1	Qualidade da decisão: A	1

Caso 008 /* Caso de crédito do tipo *hot money* à empresa X, com sucesso A */		
Diagnosticidade-Atributo	**Atributo:Valor**	**Votos-valor-atributo**
	Concessão do Crédito	
2	Caráter: A	2
1	Finalidade: A	1
2	Rendimento: A	2
2	Amortização: C	3
2	Capacidade: B	3
1	Documentação: A	1
1	Indicadores: B	2
2	Garantias: B	3
3	Nível de risco: A	14
	Resultado da Concessão	
1	Qualidade da decisão: A	1

Caso 009 /* Caso de crédito à empresa Y mediante projeto, com qualidade de decisão B */		
Diagnosticidade-Atributo	**Atributo:Valor**	**Votos-valor-atributo**
	Concessão do Crédito	
2	Caráter: A	2
1	Finalidade: B	2
2	Rendimento: B	3
2	Amortização: C	3
2	Capacidade: B	3
1	Documentação: B	2
1	Indicadores: A	1
2	Garantias: B	3
3	Nível de risco: A	20
	Resultado da Concessão	
1	Qualidade da decisão: B	2

Caso 010 /* Caso de crédito à empresa Z mediante Projeto "politicamente" indicado e decisão D */		
Diagnosticidade-Atributo	**Atributo:Valor**	**Votos-valor-atributo**
	Concessão do Crédito	
2	Caráter: B	3
1	Finalidade: D	2
2	Rendimento: B	3
2	Amortização: D	5
2	Capacidade: C	4
1	Documentação: B	2
1	Indicadores: C	3
2	Garantias: C	4
3	Nível de risco: E	42
	Resultado da Concessão	
1	Qualidade da decisão: D	3

Caso 011 /* Caso à cooperativa agro-pecuária, com decisão de qualidade B */		
Diagnosticidade-Atributo	**Atributo:Valor**	**Votos-valor-atributo**
	Concessão do Crédito	
2	Caráter: A	2
1	Finalidade: B	2
2	Rendimento: C	3
2	Amortização: C	4
2	Capacidade: B	4
1	Documentação: B	2
1	Indicadores: A	1
2	Garantias: C	4
3	Nível de risco: A	14
	Resultado da Concessão	
1	Qualidade da decisão: B	2

Caso 012 /* Caso de crédito, com sucesso B na decisão e ausência de *Garantias reais* e *Indicadores* */		
Diagnosticidade-Atributo	**Atributo:Valor**	**Votos-valor-atributo**
	Concessão do Crédito	
2	Caráter: B	3
1	Finalidade: I	2
2	Rendimento: D	4
2	Amortização: A	5
2	Capacidade: C	6
1	Documentação: B	2
1	Indicadores: ?	?
2	Garantias: ?	?
3	Nivel de risco: A	14
	Resultado da Concessão	
1	Qualidade da decisão: B	2

Caso 013 /* Caso de insucesso D na decisão e ausência do atributo *Finalidade* */		
Diagnosticidade-Atributo	**Atributo:Valor**	**Votos-valor-atributo**
	Concessão do Crédito	
2	Caráter: C	5
1	Finalidade: ?	?
2	Rendimento: D	4
2	Amortização: A	5
2	Capacidade: C	6
1	Documentação: B	2
1	Indicadores: C	3
2	Garantias: C	4
3	Nivel de risco: B	27
	Resultado da Concessão	
1	Qualidade da decisão: D	3

Caso 014 /* Caso de insucesso C, com recuperação de crédito, e ausência de atributos */		
Diagnosticidade-Atributo	**Atributo:Valor**	**Votos-valor-atributo**
	Concessão do Crédito	
2	Caráter: B	3
1	Finalidade: ?	?
2	Rendimento: D	4
2	Amortização: A	5
2	Capacidade: ?	?
1	Documentação: B	2
1	Indicadores: ?	?
2	Garantias: C	4
3	Nivel de risco: B	25
	Resultado da Concessão	
1	Qualidade da decisão: C	3

Caso 015 /* Caso de crédito à indústria, com sucesso A e ausência do atributo *Garantias* */		
Diagnosticidade-Atributo	**Atributo:Valor**	**Votos-valor-atributo**
	Concessão do Crédito	
2	Caráter: A	2
1	Finalidade: ?	?
2	Rendimento: A	2
2	Amortização: C	5
2	Capacidade: C	6
1	Documentação: B	2
1	Indicadores: C	3
2	Garantias: ?	?
3	Nível de risco: A	14
	Resultado da Concessão	
1	Qualidade da decisão: A	1

Caso 016 /* Caso de crédito a funcionário público, com decisão B e na ausência do atributo *Caráter* */		
Diagnosticidade-Atributo	**Atributo:Valor**	**Votos-valor-atributo**
	Concessão do Crédito	
2	Caráter: ?	?
1	Finalidade: A	1
2	Rendimento: D	4
2	Amortização: A	5
2	Capacidade: C	6
1	Documentação: B	2
1	Indicadores: C	3
2	Garantias: C	4
3	Nível de risco: A	14
	Resultado da Concessão	
1	Qualidade da decisão: B	2

Caso 017 /* Caso de crédito *hot money*, com insucesso D e ausência do atributo *Capacidade* */		
Diagnosticidade-Atributo	**Atributo:Valor**	**Votos-valor-atributo**
	Concessão do Crédito	
2	Caráter: C	4
1	Finalidade: A	1
2	Rendimento: D	4
2	Amortização: A	5
2	Capacidade: ?	?
1	Documentação: B	2
1	Indicadores: C	3
2	Garantias: C	4
3	Nível de risco: B	27
	Resultado da Concessão	
1	Qualidade da decisão: D	3

Caso 018 /* Caso de crédito pessoal com sucesso A na decisão e ausência de atributos */		
Diagnosticidade-Atributo	**Atributo:Valor**	**Votos-valor-atributo**
	Concessão do Crédito	
2	Caráter: ?	?
1	Finalidade: ?	?
2	Rendimento: D	4
2	Amortização: A	5
2	Capacidade: ?	?
1	Documentação: B	2
1	Indicadores: C	3
2	Garantias: C	4
3	Nível de risco: A	14
	Resultado da Concessão	
1	Qualidade da decisão: A	1

Caso 019 /* Caso de crédito com insucesso D na decisão e ausência de vários atributos */		
Diagnosticidade-Atributo	**Atributo:Valor**	**Votos-valor-atributo**
	Concessão do Crédito	
2	Caráter: A	2
1	Finalidade: ?	?
2	Rendimento: D	4
2	Amortização: A	5
2	Capacidade: ?	?
1	Documentação: A	1
1	Indicadores: ?	?
2	Garantias: ?	?
3	Nível de risco: A	14
	Resultado da Concessão	
1	Qualidade da decisão: D	3

Caso 020 /* Caso de crédito industrial, com sucesso A e presença de *Amortização* em dólar */		
Diagnosticidade-Atributo	**Atributo:Valor**	**Votos-valor-atributo**
	Concessão do Crédito	
2	Caráter: A	2
1	Finalidade: B	1
2	Rendimento: A	2
2	Amortização: C	4
2	Capacidade: A	6
1	Documentação: A	1
1	Indicadores: A	1
2	Garantias: A	2
3	Nível de risco: B	21
	Resultado da Concessão	
1	Qualidade da decisão: A	1