MARCUS SALERNO DE AQUINO

BACKFORNEB: UMA FERRAMENTA PARA CONSTRUÇAÕ DE SISTEMAS ESPECIALISTAS DIAGNOSTICADORES

Dissertação apresentada ao Curso de MESTRADO EM SISTEMAS E COMPUTAÇÃO da Universidade Federal da Paraíba, em cumprimento às exigências para obtenção do Grau de Mestre.

GIUSEPPE MONGIOVI

Orientador

A657b Aquino, Marcus Salerno de
Backforneb : uma ferramenta para construcao de sistemas especialistas diagnosticadores / Marcus Salerno de Aquino. - Campina Grande, 1987.
70 f. : il.

Dissertacao (Mestrado em Sistemas e Computacao) - Universidade Federal da Paraiba, Centro de Ciencias e Tecnologia.

1. Backforneb - 2. Sistemas Especialistas - 3. Dissertacao I. Mongiovi, Giuseppe, Prof. II. Universidade Federal da Paraiba - Campina Grande (PB) III. Título

CDU 004.891(043)

A Dona Yayá Pessôa dos Santos.

In Memoriun.

AGRADECIMENTOS

Ao meu orientador Giuseppe Mongiovi, pelo incentivo, trabalho e dedicação durante o desenvolvimento deste projeto, e que com sua perspicácia, sempre encontrou simples soluções para grandes problemas.

À Hélio de Menezes Silva, que com o seu conhecimento e experiência, muito contribuiu para a elaboração deste trabalho, e foi para mim uma imensa fonte de consulta, onde ajudou a esclarecer e definir os conceitos fundamentais desta tese.

Em especial, à minha esposa Dêlma (Baby), que sempre me acompanhou, participou, incentivou e compartilhou dos momentos de dificuldades e de realizações. Tenho certeza que sem ela este trabalho jamais teria sido realizado.

BACKFORNEB - UMA FERRAMENTA PARA CONSTRUÇÃO DE SISTEMAS ESPECIALISTAS

MARCUS SALERNO DE AQUINO

TESE SUBMETIDA AO CORPO DOCENTE DA COORDENAÇÃO DO CURSO DE PÓS-GRADUAÇÃO EM SISTEMAS E COMPUTAÇÃO DA UNIVERSIDADE FEDERAL DA PARAÍBA COMO PARTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE MESTRE EM CIÊNCIAS (M.Sc.).

Aprovado por:

GIUSEPPE MONGIOVI - M.Sc.
- Presidente -

ANTONIO EDUARDO C. PEREIRA - Dr.
- Examinador -

HÉLIO DE MENEZES SILVA - M.Sc.
- Examinador -

ULRICH SCHIEL - Dr.
- Examinador -

CAMPINA GRANDE - PB
OUTUBRO - 1987

BACKFORNEB: UMA FERRAMENTA PARA CONSTRUÇAO DE SISTEMAS ESPECIALISTAS DIAGNOSTICADORES

RESUMO

Este trabalho descreve o projeto e a implementação de uma ferramenta para desenvolvimento de sistemas especialistas diagnosticadores de uso geral (BACKFORNEB). O conhecimento é representado sob a forma de regras e a propagação da promissoriedade dos fatos na rede se fará através de encadeamento "forward" e "backward", utilizando inferência nebulosa.

BACKFORNEB consiste de três módulos principais: EDITAR, EXECUTAR e OPERAÇÕES DE I/O. O módulo EDITAR, denominado EDICON, auxilia o Engenheiro de Conhecimento a construir a Base de Conhecimento desejada de forma organizada e estruturada. O módulo EXECUTAR, é o motor de inferências do sistema e terá uma grande interação com o usuário, fazendo-lhe perguntas, recebendo respostas que foram solicitadas ou as que foram fornecidas expontaneamente e, ao final, apresentando-lhe os possíveis diagnósticos. As OPERAÇÕES DE I/O são necessárias para a boa utilização da ferramenta.

Neste trabalho são descritos também o guia de utilização do usuário e os aspectos de implementação do sistema.

SUMÁRIO

BACKFORNEB: UMA FERRAMENTA PARA CONSTRUÇAÕ DE SISTEMAS ESPECIALISTAS DIAGNOSTICADORES

1. INTRODUÇÃO

Inteligência Artificial (IA) é o ramo da ciência da computação que desenvolve conceitos e métodos que fazem uma máquina se comportar de maneira inteligente, significando não apenas a capacidade inovadora e criativa dos seres humanos mas, principalmente, como adquirir, transformar e aplicar conhecimentos.

Uma das principais aplicações de IA é o desenvolvimento de produtos cuja finalidade primordial é resolver problemas [BUCH 84].

1.1 SISTEMAS ESPECIALISTAS

Sistemas Especialistas (SE's) são programas que solucionam problemas substanciais, geralmente reconhecidos como difíceis e requerendo perícia. São chamados de "sistemas baseados no conhecimento" porque sua performance depende criticamente do uso de fatos e heurísticas usados pelos especialistas [STEF 82].

Muitos destes sistemas têm a capacidade de justificar o seu raciocínio, possibilitando ao usuário fornecer respostas mais precisas às perguntas, e aumentar a confiança nos resultados. Posssibilita também ao especialista, checar a consistência das regras, e a encontrar erros no conhecimento caso eles ocorram [BARR 81].

O principal objetivo de um SE é de atingir um alto grau de desempenho na realização de uma tarefa. O SE tenta imitar, até certo ponto, a maneira como um especialista toma decisões, procurando produzir resultados de alta qualidade em um curto espaço de tempo [SAND 85].

Os SE's têm sido muito usados principalmente na área de Medicina (MYCIN - ajuda a diagnosticar/sanar infecções bacterianas), de Geologia (PROSPECTOR - ajuda geologistas a avaliarem o potencial mineral de uma região), de Quimica (DENDRAL - primeiro SE construído com sucesso, auxilia na determinação da estrutura molecular de um composto quimico, utilizando-se de informações contidas na espectometria de massa e ressonância magnética); são utilizados também em Sistemas Computadorizados (XCON - configura computadores VAX - 11/780), Engenharia (REACTOR - ajuda a diagnosticar/sanar acidentes em reatores nucleares), etc.

A Figura 1.1 apresenta uma estrutura geral de um SE que tem como grande vantagem, em relação aos sistemas convencionais, a Base de Conhecimento (BC) ser separada da estratégia de contrôle.

A construção da BC é feita pelo Engenheiro do Conhecimento (EC), no qual ele transfere todas as informações consideradas relevantes pelo especialista, para a resolução do problema. A manipulação da BC através de um Sistema de Contrôle, permite ao SE inferir respostas e soluções do problema, como também justificar suas ações. Um sistema de interface com o

usuário possibilita o conhecimento de fatos que auxiliam a busca de soluções, a modificação da BC e a divulgação dos resultados atingidos pelo sistema.

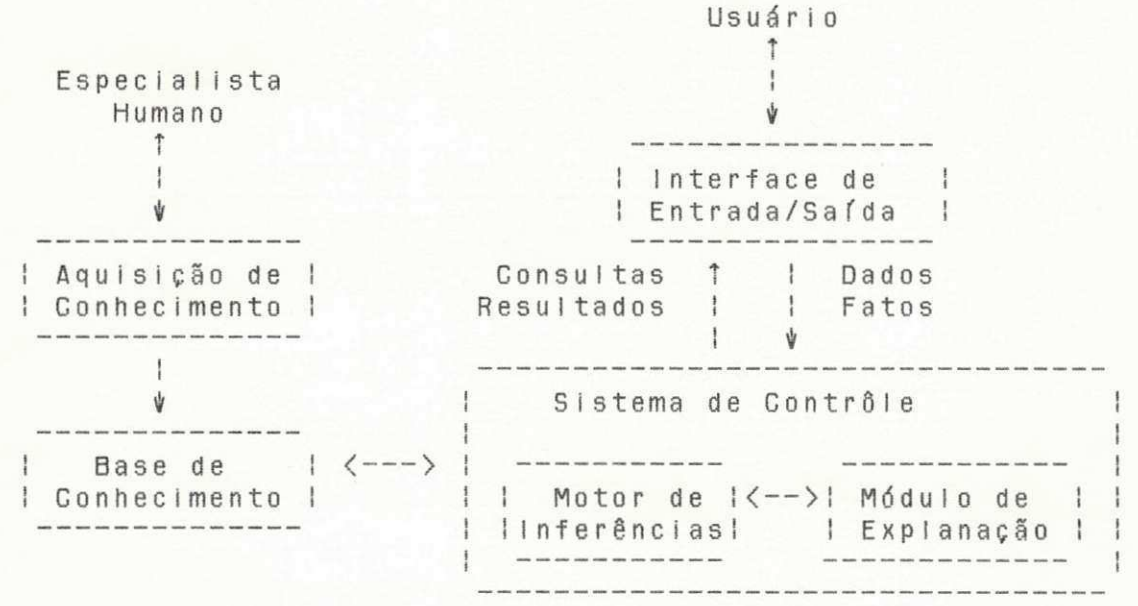


Figura 1.1 - Estrutura de um SE

SE's têm sido desenvolvidos para solucionar vários tipos diferentes de problemas, mas suas atividades básicas podem ser agrupadas em seis categorias que são [STEF 82]:

1) **Interpretação:** envolve a análise de dados para determinar o seu significado (ex: interpretação dos dados de um espectômetro de massa [BUCH 78]).

2) **Diagnóstico:** é o processo de encontrar falhas/defeitos em um sistema baseado na interpretação dos fatos observados (ex: diagnóstico de doenças infecciosas [SHOR 76]).

3) **Monitoração:** imterpreta continuamente sinais e ativa alarme quando for necessária uma intervenção (ex: monitoramento de um paciente utilizando pulmão artificial [FAGA 80]).

4) **Predição:** significa determinar o futuro a partir de modêlos do passado e do presente (ex: predição dos efeitos de uma mudança na polítca econômica).

5) **Planejamento:** programa de ações que podem ser utilizados para atingir seus objetivos (ex: planos experimentais em engenharia genética [STEF 81]).

6) **Projeto:** estabelecimento de especificação para criar objetos que satisfaçam requisitos específicos (ex: projetar um circuito digital [McDE 80])

1.2 FERRAMENTAS PARA DESENVOLVIMENTO DE SISTEMAS ESPECIALISTAS

As ferramentas são importantes porque simplificam o trabalho de construção de SE's, em particular a extração do conhecimento de especialistas humanos e sua posterior representação.

Podem ser classificadas basicamente em três grupos, que são [HAYE 83]:

- **Sistemas do tipo "shell":** Estes sistemas derivam de SE's já existentes, pois removem a sua BC e aproveitam apenas o motor de inferências e suas facilidades de suporte. Os sistemas do tipo

"shell" provêem estruturas e facilidades de construção que torna o desenvolvimento de SE's mais fácil e mais rápido. Mas por outro lado, perdem em generalidade e flexibilidade pois podem trabalhar somente em uma restrita classe de problemas [WATE 85]. Os "shell's" mais conhecidos são o EMYCIN (derivado do MYCIN) [MELL 74], KAS (derivado do PROSPECTOR) [DUDA 81], EXPERT (derivado do CASNET) [WEIS 79].

- **Linguagens de representação de propósito geral:** São linguagens desenvolvidas especificamente para o EC e podem ser aplicadas em várias áreas do conhecimento. Provêm maior controle no acesso aos dados e na inferência do que os sistemas do tipo "shell", mas também são mais difíceis de usar. Como exemplo desse tipo de ferramentas temos a linguagem ROSIE [FAIN 82], OPS5 [FORG 81], RLL [GREI 80], HEARSAY III [ERMA 81].

- **Ferramentas de apoio a construção de projetos:** Consistem de programas que podem ajudar a obter e a representar o conheciemnto e projetar a implementação de SE's. Os sistemas mais conhecidos são o AGE (apoio a projetos) [NII 79] e TEIRESIAS (aquisição de conheciemnto) [DAVI 76].

1.3 POR QUE BACKFORNEB?

Com a crescente explosão e impacto no uso de SE's, torna-se necessário o desenvolvimento de ferramentas que tanto venham, cada vez mais, auxiliar na construção, desenvolvimento e aplicações desses sistemas, bem como ajudar a extrair e

representar o conhecimento dos especialistas humanos de forma eficiente e amigável.

Para o desenvolvimento de um SE, uma primeira alternativa seria desenvolvê-lo sob medida, ad-hoc, a partir do nada, em uma linguagem a nível de Pascal ou, melhor ainda, de PROLOG. No entanto, isto seria uma maneira pouco proveitosa do uso do tempo dos programadores, que ao desenvolverem um SE, pouco aproveitariam dos SE's previamente desenvolvidos. Uma segunda alternativa seria usarmos um "shell" e acrescentar-lhe o conhecimeto necessário ao novo domínio de aplicação. Todavia, como os "shells" são em geral demasiadamente rígidos, este processo torna-se não natural e improdutivo, exceto para domínios muito próximos daqueles que os originaram. Uma terceira alternativa é o desenvolvimento de um sistema geral para cada classe de SE's (diagnosticadores, interpretadores, monitoradores, etc.).

Dentro desse espírito, vimos a necessidade de se criar um sistema que possuísse um motor de inferências eficiente integrado a um editor de conhecimento. Para confirmar a necessidade dessa ferramenta, os SE's SINDROMUS [NICO 87] e OFTALMO [CHIA 86], em suas conclusões, ressaltam a necessidade de um editor mais flexível para validação das respectivas BC's.

Portanto, o objetivo deste trabalho é construir uma ferrramenta (**BACKFORNEB** - encadeamento **BACK**ward/**FOR**ward com Inferência **NEB**ulosa) para desenvolvimento de Sistemas Especialistas diagnosticadores de uso geral em microcomputadores

com interface amigável ao engenheiro do conhecimento e ao usuário final, onde cada SE específico será construído pelo fornecimento da respectiva base de conhecimento.

O Capítulo 2, contém a descrição do sistema BACKFORNEB, como é representado o seu conhecimento, o tipo de inferência utilizada pelo sistema e as suas características básicas.

O Capítulo 3 contém o guia de utilização do usuário, descrevendo uma sessão de como funcionam o EDICON (Editor de Conhecimento) e o Motor de Inferências.

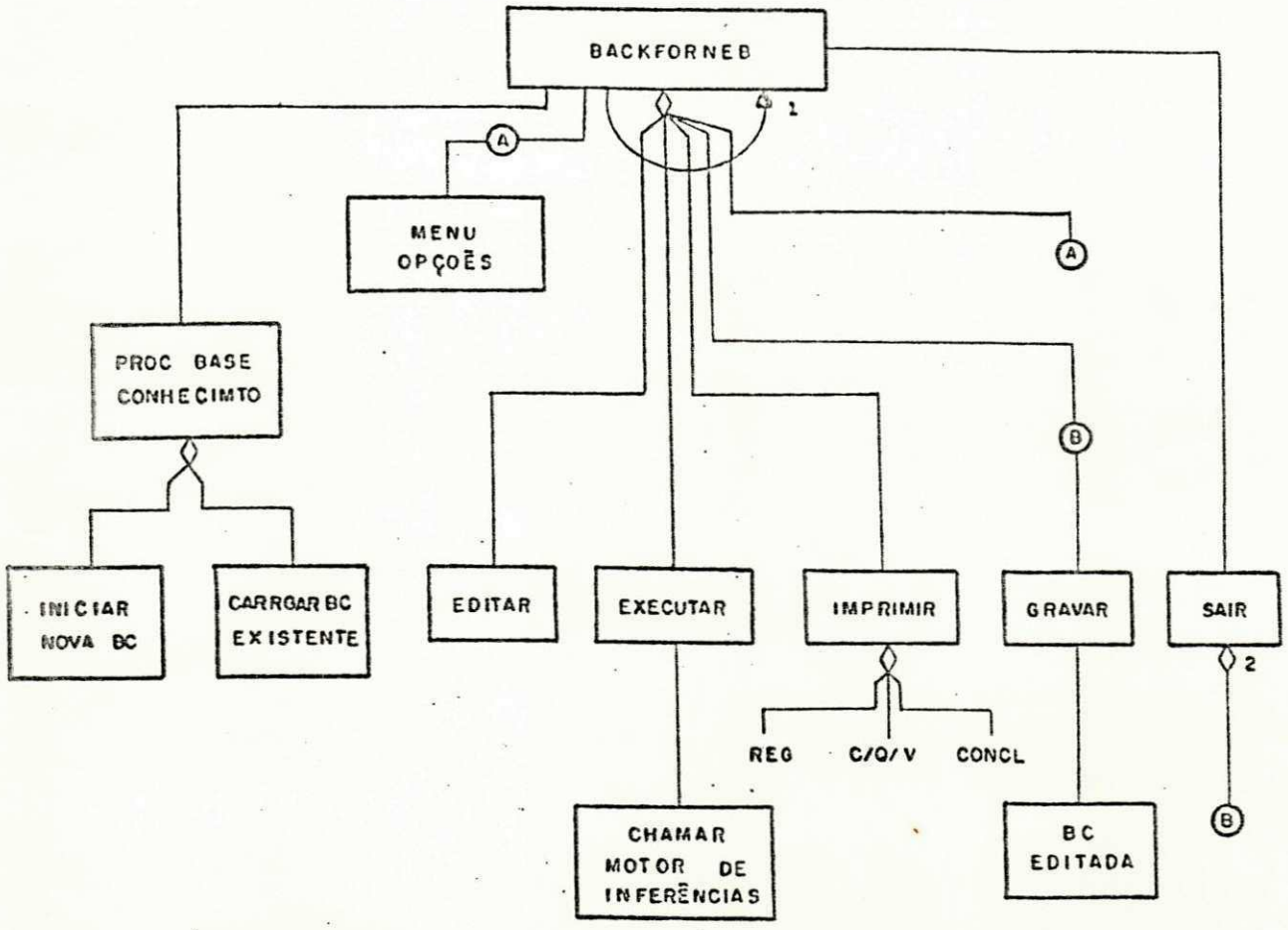
Os aspectos de implementação de BACKFORNEB são descritos no Capítulo 4, mostrando as estruturas de dados, os gráficos estruturados de seu funcionamento e os algorítmos básicos principais.

Finalmente, no Capítulo 5 são apresentados a conclusão e sugestões para futuros trabalhos. É também feita uma avaliação crítica em relação a sua capacidade e a sua utilização (flexibilidade, tempo de resposta, e outras vantagens).

2. DESCRIÇÃO DO SISTEMA BAKFORNEB

2.1 INTRODUÇÃO

BACKFORNEB é uma ferramenta integrada para construção de Sistemas Especialistas diagnosticadores de uso geral e foi projetada conforme a Figura 2.1:



REG - LISTAR REGRAS
C/Q/V - LISTAR CLASSES/QUALIFICADORES/VALORES
CONCL - LISTAR TODAS AS CONCLUSÕES

1. ENQTO OPCÕES ≠ SAIR
2. VERIFICAR SE USUÁRIO QUER GRAVAR TODOS OS ARQUIVOS OU ABANDONAR

MENU OPÇÕES: (EDITAR, EXECUTAR, IMPRIMIR, GRAVAR, SAIR)

Figura 2.1 - BACKFORNEB - Uma Ferramenta para Construção de SE's

Os módulos principais desta ferramenta são:

- **EDITAR**: é o editor de conhecimentos (EDICON) que permite o EC editar as regras da BC desejada.

- **EXECUTAR**: é o motor de inferências do sistema e terá uma grande interação com o usuário, fazendo-lhe diversas perguntas, recebendo as respostas que lhe solicitou e as que ele forneceu expontaneamente e, ao final, apresentando-lhe os possíveis diagnósticos.

- **OPERAÇÕES I/O** (Carregar BC, Gravar BC, Listar Regras): são necessárias para a boa utilização da ferramenta.

2.2 DESCRIÇÃO DO EDICON - UM EDITOR DE CONHECIMENTOS

O ponto de estrangulamento ou gargalo, no desenvolvimento de um SE, é a aquisição de conhecimento, pois envolve longa e difícil interação com peritos [AQUI 87]. A obtenção do conhecimento pode englobar três funções:

1) Edição (isto é: inserção, eliminação, modificação, etc.) do conhecimento;

2) Checagem da consistência da BC como um todo, antes de aceitar qualquer edição;

3) Indução, generalização, ou particularização automática de regras (ou outros formalismos) de conhecimento.

Atualmente somente a primeira função acima tem sido

implementada em pacotes comerciais, sendo as outras duas ainda objetos de pesquisas.

A edição da BC, sem uma ferramenta apropriada, é uma das tarefas mais espinhosas na construção de SE's. Isto deve-se ao fato de que constantemente é necessário adicionar, excluir, e modificar informações válidas à modelagem do conhecimento. Entretanto, a disponibilidade de um editor específico, amigável e confiável, pode tornar esta tarefa muito mais fácil.

Obviamente é necessário que um editor ofereça facilidades para lidar com o conhecimento bem estruturado e fácilmente modificável. É necessário também que esse editor tenha uma boa interação com o EC, seja bastante flexível e de uso mais geral possível.

O EDICON auxilia o EC a construir sua BC de forma organizada e estruturada. Tem facilidades para inserir, eliminar e modificar regras, e também faz com que o EC forneça gradativamente, as regras que possuam a mesma conclusão, dentro da mesma classe e do mesmo qualificador. Este procedimento é importante para que as regras sejam fornecidas agrupadas dentro de um mesmo conhecimento.

2.3 ESTRUTURA DO CONHECIMENTO

Uma BC pode ter o conhecimento expresso de vários modos:

- Regras ("rule based systems") : são representadas na forma de "Se condição ocorre, então faça esta ação";

- **Redes Semânticas:** consiste de nodos representando objetos, conceitos ou eventos, ligados entre si por arcos que representam sua relação;

- **Quadros ("Frames"):** representam uma estrutura de dados que inclui informações declarativas e procedurais em suas relações pré-definidas [BARR 81].

Dentre as formas acima citadas, optou-se por formalizar a BC de BACKFORNEB utilizando-se regras, pois estas possuem uma estrutura próxima da linguagem natural, são de fácil entendimento e de fácil processamento. Cabe salientar que várias ferramentas já usam este tipo de representação (ex: EXSYS [EXSY 86], VPX [VPX 86]).

2.3.1 Descrição das regras

Uma regra, para o EDICON, é definida como:

SE: <CONDIÇÕES>
ENTÃO: <CONCLUSÃO>
FATOR DE ATENUAÇÃO: FA

onde:

<CONDIÇÕES> - é um fato, ou um NOT fato, ou uma sequência de fatos separadas por um mesmo operador lógico (conectivo: AND, OR, XOR, INDEP);

<CONCLUSÃO> - é apenas a declaração de um fato;

<FATOR DE

ATENUAÇÃO> - corresponde a chance de ocorrência da hipótese (<CONCLUSÃO>) caso ela seja confirmada (no nosso sistema, o FA pode variar entre 0.01 a 1.00).

Os fatos podem ser descritos através de sentenças da linguagem natural (ex: O terminal de video é do tipo gráfico). Como muitos fatos relacionados, normalmente tem em comum um mesmo núcleo (prefixo da frase) e diferem apenas no seu complemento (sufixo da frase), adotamos a seguinte estrutura para representá-los:

<FATO> => <QUALIFICADOR> <VALOR>

onde:

<QUALIFICADOR> - prefixo da frase, normalmente terminado por um verbo (ex: O terminal de video é).

<VALOR> - é o complemento do qualificador, normalmente um objeto (ex: do tipo gráfico).

Portanto um exemplo de uma regra será:

SE: **O destino das aplicações é** desenvolvimento de software

&

O destino das aplicações é sistema especialista

&

O software a ser desenvolvido é para uso gráfico

ENTÃO:

Terminal de video é do tipo gráfico

FATOR DE ATENUAÇÃO = 0.90

O acesso a um qualificador poderia ser feito via um número de identificação ou citando expressamente seu nome por extenso. Como os SE's reais normalmente lidam com um grande número de qualificadores, este tipo de acesso dificultaria bastante a sua localização.

Para contornar esta dificuldade, optamos por dotar o EDICON de facilidades que permitam ao usuário agrupar os qualificadores em classes distintas. Esta estrutura permite que, a qualquer instante, o usuário possa localizar facilmente as classes existentes e os qualificadores dentro de cada classe. Isto, além da facilidade acima mencionada, permite ao usuário criar uma base de conhecimento melhor estruturada.

2.3.2 Tipos de Conectivos

Entre os conectivos utilizados, AND, OR, NOT e XOR têm o mesmo significado da álgebra booleana, enquanto que o conectivo IND tem um significado especial. Se tivermos duas <CONCLUSÕES> com o mesmo nome, ou seja:

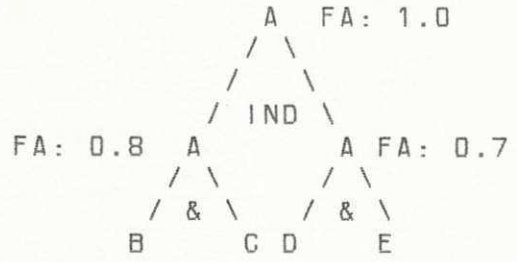
1. A <-(0.8)- B & C
2. A <-(0.7)- D & E

O EDICON criará automaticamente, uma nova regra (regra 3) que será composta de um nó pai (Conclusão A) e cujos filhos serão as Conclusões das regras 1 e 2. Teremos então:

1. A <-(0.8)- B & C

2. A <-(0.7)- D & E

3. A <-(1.0)- Concl1 IND Concl2



A regra 3 é importante porque a promissoriedade da regra 1 aumenta se a regra 2 for provada, e vice-versa. Portanto, todas as regras que referenciarem a regra 1 (ou 2) irão agora referenciar a regra 3. As regras que forem adicionadas à rede com mesma Conclusão da regra 1, serão agora consideradas como filhas da regra 3.

2.4 DESCRIÇÃO DO MOTOR DE INFERÊNCIAS

BACKFORNEB tem seu conhecimento representado internamente sob a forma de um grafo bidirecionado, utilizando inferência nebulosa (Fuzzy Inference) do tipo do MYCIN [FORS 84], com encadeamento "forward" e "backward". Para uma denominação mais coerente, chamaremos nossa estrutura de Rede de Inferências de acordo com [WATE 85].

Após a edição da BC, EDICON fornece como resultado uma rede de inferências como mostra a Figura 2.2.

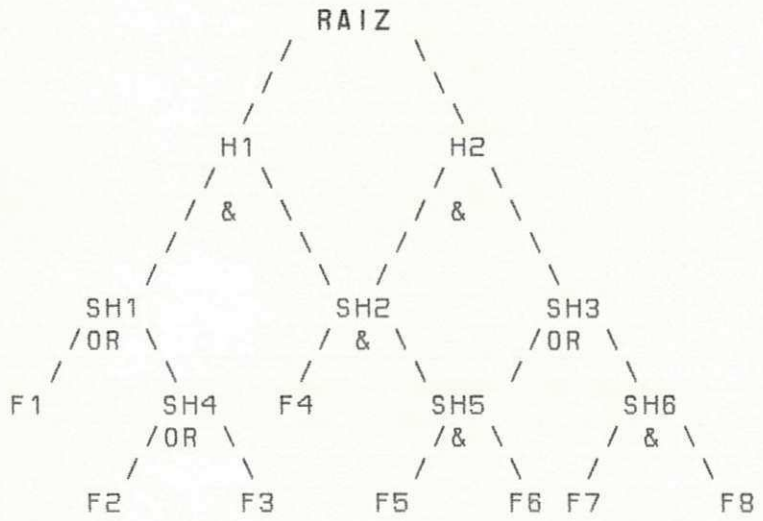


Figura 2.2 - Exemplo de uma Rede de Inferências

Nesta rede os nodos Hi's representam as hipóteses a serem provadas (nodos sem pai) e as folhas são os fatos que podem ser voluntariados.

Para que, durante a execução, o sistema forneça a lista de hipóteses promissoras, é necessário que na edição, essas hipóteses sejam de alguma forma agrupadas ou facilmente acessadas.

Como essas hipóteses sempre são o topo de suas respectivas sub-árvores de conhecimento, optou-se por unî-las em uma única árvore através da criação de um nodo chamado RAIZ cujos filhos serão todas as hipóteses existentes.

2.4.1 Inferência Nebuloza (Fuzzy Inference)

Inferência Nebulosa foi criada por Lotfi Zadeh (1965) que estendeu a lógica booleana para os números reais [FORS 84]. Na Algebra Booleana, 1 representa verdadeiro e 0 falso. Isto existe também na Inferência Nebulosa, mas além disso, todos os reais entre 0 e 1 são empregados para indicar verdade parcial. Por exemplo:

p(alto(x)) = 0.75

Esta função indica que "x é alto" é três quartos verdadeira, e da mesma forma, um quarto falsa. Para combinar valores verdade, a Inferência Nebulosa define o equivalente aos operadores lógicos AND, OR e NOT:

```
p1 AND p2 = MIN(p1,p2)      (* menor *)
p1 OR  p2 = MAX(p1,p2)      (* maior *)
NOT p1    = 1 - p1          (* inverso *)
```

Utilizando esta teoria BACKFORNEB foi estruturado da seguinte forma:

Seja a regra, **C <-(FA)-- C1 op C2 op ... op Cn**

onde **C** é a conclusão, **Ci's** (i = 1,..,n) são as Condições, **FA** é o fator de atenuação e **op** é um operador lógico (conectivo) pertencente ao conjunto {AND, OR, XOR, IND}. A cada Ci está associado um grau de certeza **GCi**, e **GC** é o grau de certeza de C.

Para cada conectivo temos:

- Conectivos AND e OR

Para estes conectivos GC é dado por:

GC = min(GC1,GC2,...,GCn) * FA para o conectivo AND

e

GC = max(GC1,GC2,...,GCn) * FA para o conectivo OR

- Conectivo XOR

Para este conectivo, quando uma condição Ci for considerada (assinalada) implica que as demais não são mais voluntariáveis (isto é, elas são descartadas para futuras consultas). A propagação do grau de certeza, neste caso, se fará automaticamente do GCi para GC, observando-se o FA. Isto é,

GC = GCi * FA

O grau de certeza das demais Condições, será tomado como igual ao GCmin (grau de certeza mínimo), o que faz com que elas não sejam mais perguntáveis.

- Conectivo IND

As características do nodo IND são Ci = Ci-1 para i = 2,...,n e FA = 1. Shortliffe quando desenvolveu o MYCIN, utilizou para o conectivo IND a seguinte fórmula:

p = p1 + (1 - p1) * p2 (1)

Onde p1 e p2 são as probabilidades das regras C1 e C2 ocorrerem respectivamente e p a probabilidade de C. Como o intervalo de probabilidade, neste caso, varia entre 0 e 1, e no caso de BACKFORNEB é de GCmin a GCmax, utiliza-se uma fórmula para converter os valores de p para GC,

$$p = (GC - GCmin) / (GCmax - GCmin) \qquad (2)$$

Substituindo-se (2) em (1) temos:

$$\frac{(GC - GCmin)}{(GCmax-GCmin)} = \frac{(GC1 - GCmin)}{(GCmax-GCmin)} + \left[1 - \frac{(GC1 - GCmin)}{(GCmax-GCmin)}\right] * \frac{(GC2-GCmin)}{(GCmax-GCmin)}$$

Simplificando, temos:

$$GCa = GC1 + [(GC2 - GCmin) + (GCmax - GC1)] / (GCmax - GCmin)$$

Se caso existir mais Ci's em um mesmo nodo independente, atribui-se a GC1 o valor encontrado em GCa, e ao valor do próximo Ci à GC2, e novamente é aplicada, recursivamente para todos os Ci's, a fórmula descrita.

- Conectivo NOT

Para este conectivo a regra toma a forma

C <-(FA)-- NOT C1

Sendo p e p1 as probabilidades associadas a C e C1 respectivamente, temos:

$$p = NOT\ p1 = 1 - p1 \quad \text{(considerando FA = 1)}$$

em termos de grau de certeza, a expressão acima ficaria,

$$\frac{(GC - GCmin)}{(GCmax - GCmin)} = 1 - \frac{(GC1 - GCmin)}{(GCmax - GCmin)}$$

simplificando a expressão temos,

$$GC = GCmax + GCmin - GC1$$

considerando o FA temos,

$$GC = (GCmax + GCmin - GC1) * FA$$

Para ilustrar a propagação dos graus de certeza, consideremos a regra C <-(FA)-- C1 op C2 op C3, GCmin = -5 e GCmax = +5. O resultado é o mostrado na tabela da Figura 2.3:

	GC1	GC2	GC3	FA	GC
AND	3	5	2	0.90	1.80
OR	3	5	2	0.90	4.50
NOT	3	-	-	0.90	-2.70
XOR	-5	-5	2	0.90	1.80
IND	3	1	2	1.00	4.70

Figura 2.3 - Propagação dos Graus de Certeza

2.4.2 Encadeamento Forward e Backward

Para cada fato voluntariado pelo usuário, o respectivo GC é automaticamente propagado por toda a rede até atingir as hipóteses. Isto se chama encadeamento forward (Fatos -> Hipótese).

Caso o usuário deseje provar uma hipótese, o sistema irá procurar (fazendo uma análise em encadeamento backward (Hipótese -> Fatos) quais são os fatos dessa hipótese, ainda não assinalados, necessários para a confirmação/desconfirmação da hipótese.

Seja o exemplo mostrado na Figura 2.5:

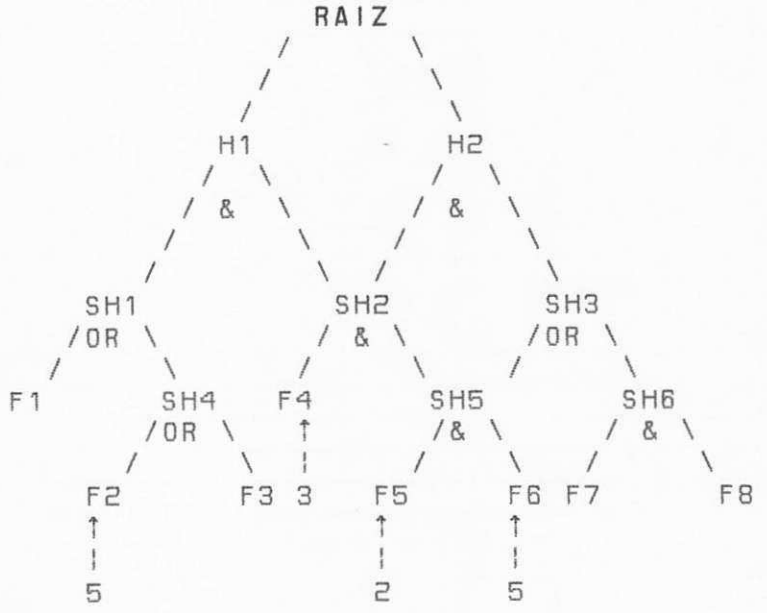


Figura 2.5 - Exemplo da propagação dos GC's

Supondo que sejam voluntariados os fatos F2, F4, F5 e F6 com os GC's 5, 3, 2 e 5, respectivamente, tem-se os resultados na Figura 2.6 (os fatos não voluntariados possuem um GC inicial, que chamaremos GCInic, cujo valor está fora do intervalo GCmin a GCmax):

	Conect	FA	GC
H1	&	0.90	1.30
H2	&	0.90	1.15
SH1	OR	0.80	2.80
SH2	&	0.90	1.44
SH3	OR	0.80	1.28
SH4	OR	0.70	3.50
SH5	&	0.80	1.60
SH6	&	0.95	GCInic
F1	-	-	GCInic
F2	-	-	5
F3	-	-	GCInic
F4	-	-	3
F5	-	-	2
F6	-	-	5
F7	-	-	GCInic
F8	-	-	GCInic

Figura 2.6 - Propagação dos GC's na Rede de Inferências

Neste exemplo, as hipóteses H1 e H2 têm uma promissoriedade de 1.3 e 1.15 respectivamente. Para se provar a hipótese H2, BACKFORNEB irá perguntar então, qual o GC de F7 e F8. Obtida a resposta, o sistema novamente irá propagar (agora em encadeamento forward) os respectivos GC's pela rede e fornecer as novas promissoriedades de H1 e H2.

2.4.3 Características Básicas do Motor de Inferências

Algumas características importantes que o sistema BACKFORNEB apresenta são as seguintes:

a) O sistema não possui um diagnóstico único. BACKFORNEB apresenta a lista de hipóteses em ordem decrescente de promissoriedade com seus respectivos GC's. Isto é importante, para se dar uma visão mais ampla do raciocínio empregado pelo sistema, e também para que o usuário possa escolher a hipótese (para ser provada) que lhe pareça mais urgente ou mais importante. Note que a decisão final do diagnóstico é sempre do usuário, BACKFORNEB tem como função apenas a de auxílio ao diagnóstico.

b) O voluntariamento de um fato, bem como a mudança de seu GC, pode ser feito em qualquer ponto da consulta. Isto é importante porque o usuário pode, com o decorrer das perguntas que o sistema lhe fará, informar algum fato novo (ainda não voluntariado), e assim agilizar a consulta.

c) BACKFORNEB sempre guarda a última consulta do Diagnosticando (Dgando). Quando o usuário quiser reativar uma consulta suspensa, o sistema carrega apenas os GC's do Dgando (armazenados em um arquivo denominado "Dgando.GC"), restaurando-os à rede de inferências (que é invariável).

CAPÍTULO 3 - GUIA DE UTILIZAÇÃO

O sistemas BACKFORNEB ao ser ativado apresenta a tela abaixo, perguntando ao usuário qual o SE que ele deseja usar.

```
==========================================================================
QUAL O NOME DO SISTEMA ESPECIALISTA (MAX 8 CARACTERES)?
U:
==========================================================================

BEM VINDO AO SISTEMA BACKFORNEB

DEPARTAMENTO DE SISTEMAS E COMPUTACAO
CENTRO DE CIENCIAS E TECNOLOGIA
UNIVERSIDADE FEDERAL DA PARAIBA
SISTEMA DESENVOLVIDO POR:
            MARCUS SALERNO DE AQUINO
            GIUSEPPE MONGIOVI
            HELIO DE MENEZES SILVA

                    OUTUBRO/87

==========================================================================
```

Após o fornecimento do nome do SE desejado, BACKFORNEB irá carregar a respectiva BC caso ela exista, caso contrário, fornecerá uma mensagem indicando que se trata de um novo SE a ser criado. Em ambos os casos, após apresentar a mensagem adequada, mostrará a seguinte tela:

```
==========================================================================
SISTEMA ESPECIALISTA  *** Nome da BC ***
OPCOES: ED[i]TAR, e[x]CUTAR, IM[p]RIMIR, [g]RAVAR, [Esc]SAIR:
==========================================================================
```

```
==========================================================================
```

Neste momento o usuário poderá:

ED[i]TAR - modificar a BC corrente ou editar uma nova BC. Para isso, o sistema irá ativar o EDICON;

E[x]CUTAR - executar a BC, ou seja, será ativado o Motor de Inferências;

IM[p]RIMIR - imprimir todas as regras, classes, qualificadores ou valores;

[g]RAVAR - gravar as regras editadas;

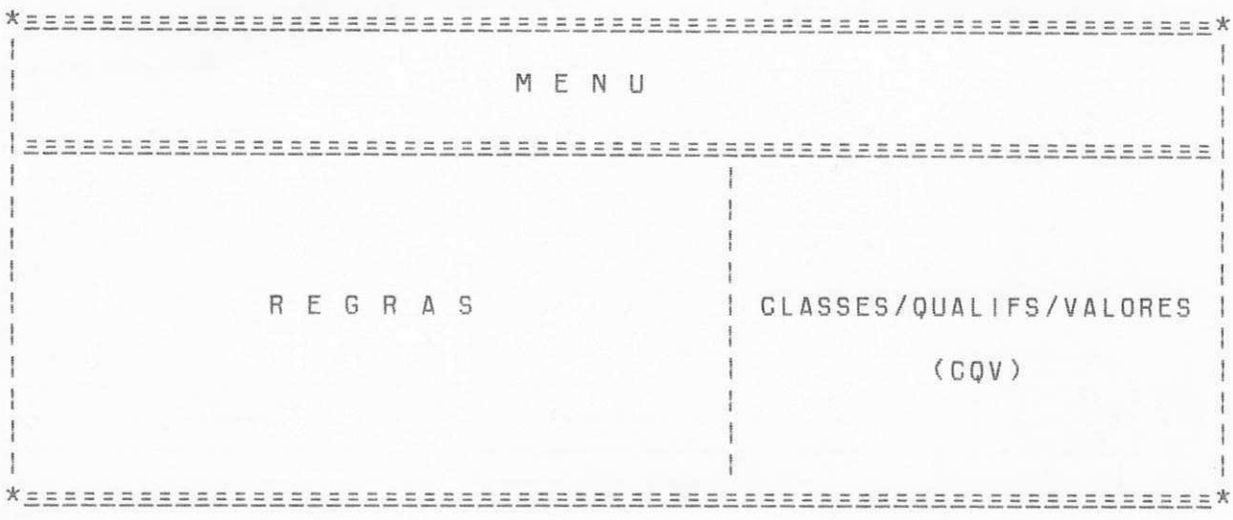
[Esc]SAIR - sair do sistema e encerrar sua utilização.

3.1 UTILIZAÇÃO DO EDICON

Nesta seção será apresentada a organização da tela de comunicação do EDICON (interface com o usuário) e o conceito de hierarquização de regras. Será apresentada também uma sessão exemplificando como o EDICON interage com o usuário.

3.1.1 Apresentação da Tela de Comunicação

A tela de comunicação com o usuário está dividida em três campos, organizados da seguinte forma:



MENU - neste campo são apresentadas as opções que o sistema tem para a situação corrente, bem como as mensagens de erros.

REGRA - este campo mostra a regra que está sendo editada.

CLASSES/QUALIFS/VALORES - este campo é utilizado para apresentar as classes, qualificadores, e valores existentes.

3.1.2 Hierarquização de Regras

O EC ao formar a sua BC pode ter, dentro de uma regra, vários fatos pertencentes a uma mesma classe. Pode ter também algumas evidências que tenham o mesmo Qualificador, por ex: O

destino das aplicações é desenvolvimento de software, e O destino das aplicações é sistema especialista. Para facilitar e organizar a edição, foi criado uma estrutura hierárquica para inserção de regras. O EC deve fornecer seguidamente todas as regras com os mesmos valores dentro dos mesmos qualificadores dentro das mesmas classes para uma conclusão (*). Por exemplo:

```
1. SE ..... ENTÃO Classe1 Qual1 Val1   FA = 0.90
2. SE ..... ENTÃO Classe1 Qual1 Val2   FA = 0.85
   .
   .
   .
8. SE ..... ENTÃO Classe1 Qual8 Val3   FA = 0.80
9. SE ..... ENTÃO Classe1 Qual8 Val12  FA = 0.95
   .
   .
   .
13.SE ..... ENTÃO Classe9 Qual7 Val5   FA = 0.60
```

Do mesmo modo, para cada regra, o EC deve fornecer seguidamente todas as condições com os mesmos qualificadores dentro das mesmas classes. Por exemplo:

(*) Note que, o sistema não necessariamente obriga o EC a fornecer as regras estruturadamente.

SE

Classe1 Qual1 Val1

& Classe1 Qual1 Val2

& Classe1 Qual1 Val3

& Classe1 Qual5 Val1

& Classe1 Qual5 Val4

& Classe8 Qual3 Val1

ENTÃO . . .

Este conceito, embora aparentemente torne o editor mais rígido, "força" a edição do conhecimento de maneira estruturada.

3.1.3 Funcionamento do EDICON

O EDICON é ativado pelo Sistema Diagnosticador BACKFORNEB sempre que o usuário desejar criar ou modificar uma BC. Neste momento, o nome da BC bem como as informações necessárias já foram obtidas por BACKFORNEB.

O EDICON, sendo ativado, apresenta a tela com o seguinte formato:

```
================================================================================
OPERAR:  <num>SELECAO, [i]NSERIR, [e]LIMINAR, [m]OVER, [a]LTERAR
REGRAS   [PagDown]AVANCA PAGINA, [PagUp]RETORNA PAGINA, [Esc]SAIR:
================================================================================
AS REGRAS DO SISTEMA SAO

 1. Um grande espaco em disco e' necessario
 2. O equipamento necessario e' IBM PC-XT
 3. Driver necessario e' p/ disco de face dupla
 4. O terminal de video e' do tipo texto
 5. O equipamento necessario e' IBM PC-XT
 6. O equipamento necessario e' IBM PC-XT
     (Combinacao das Regras: 2 5 33 76 85)
 7. O terminal de video e' do tipo texto
 8. O terminal de video e' do tipo grafico
 9. O terminal de video e' do tipo texto
     (Combinacao das Regras: 4 7 22 45)
10. O display e' monocromatico
11. A impressora e' qualidade carta
================================================================================
```

As partes <CONCLUSÃO> de todas as regras do sistema são mostradas numeradas no campo REGRAS. Se não houver nenhuma regra, o sistema apresenta uma mensagem dizendo "BASE DE CONHECIMENTO VAZIA, NÃO EXISTEM REGRAS". Note que a regra 6 (por exemplo) indica que as regras 2, 5, 33, 76, 85, possuem a mesma <CONCLUSÃO>. Esta regra, chamada de "Nó Independente" será importante na avaliação da promissoriedade das hipóteses durante a inferência.

O EC tem várias opções para trabalhar com as regras:

<num>: digitando-se o número da regra, esta será apresentada integralmente (<CONCLUSÃO> + Conectivo + <CONDIÇÕES> + F. ATENUAÇÃO);

[i] : inserir uma sequência de novas regras;

[e] : eliminar uma sequência de regras;

[m] : através desta opção pode-se ordenar ou agrupar as regras convenientemente;

[a] : se uma regra não estiver correta, o EDICON lhe dá a facilidade de corrigí-la;

[PagDown] : este comando faz avançar para a próxima página, ou seja, quando o número de regras existentes não couber na tela, esta opção faz listar as regras da próxima página;

[PagUp] : esta opção faz voltar à página anterior das regras;

[Esc] : sai deste nível.

Se o usuário desejar inserir uma regra, ele deve entrar com a opção "i". A seguir, se houver regras, o EDICON pergunta ao EC qual o número da regra que deseja inserir ("u" será para inserir após a última regra). Se não houver regras, o sistema (por "default") irá inserir a primeira regra. Após a escolha, o sistema apresenta uma nova tela:

```
==========================================================================
OPERAR : <num>SELECAO, [i]NSERIR, [e]LIMINAR, [m]OVER, [a]LTERAR
CLASSES   [Esc]SAIR:
==========================================================================
    REGRA NRO: 36                          | AS CLASSES ATUAIS SAO
                                           |
                                           | 1.APLICACAO
                                           | 2.HARDWARE
                                           | 3.CUSTOS
                                           | 4.SOFTWARE
                                           |
                                           |
                                           |
                                           |
                                           |
                                           |
                                           |
==========================================================================
```

O EDICON fica aguardando a escolha de uma classe para começar a formar a <CONCLUSÃO> da regra. Isto é feito, entrando com o número da classe desejada (opção "<num>"). Note que as opções apresentadas têm a mesma função que as opções de operar regras, mas agora estas opções são utilizadas para operar classes.

Escolhida a classe, é apresentada nova tela com a lista dos qualificadores respectivos à classe escolhida, e o mesmo conjunto de opções, mas agora utilizadas para operar qualificadores.

```
==========================================================================
OPERAR : <num>SELECAO, [i]NSERIR, [e]LIMINAR, [m]OVER, [a]LTERAR
QUALIFS  [Esc]SAIR:
==========================================================================
    REGRA NRO: 36                        | CLASSE: APLICACAO
                                         |
CLASSE: APLICACAO                        |1.Destino das aplicacoes e'
                                         |2.Maior arquivo a criar e'
                                         |3.O computador e'
                                         |4.Os arquivos a criar sao
                                         |5.Terminal de video e'
                                         |6.Impressora tipo carta e'
                                         |7.Os graficos gerados sao
                                         |8.Os dados graficos serao
                                         |
                                         |
                                         |
==========================================================================
```

Da mesma forma, escolhido um qualificador, será mostrada uma tela para operar valores. O usuário terá então que escolher um dentre uma lista de valores correspondentes ao qualificador escolhido.

Note que toda vez que o EC quiser formar uma <CONCLUSÃO> (ou <CONDIÇÃO>), ele sempre terá que escolher uma classe, seguido

de um qualificador, e seguido de um valor.

Terminada a edição da <CONCLUSÃO>, EDICON irá perguntar qual o conectivo que será usado.

```
==========================================================================
QUAL O CONECTIVO A SER USADO NA REGRA?
[o]ou, [e]&, [n]nao, [x]ou-exclusivo, [Esc]sair:
==========================================================================
    REGRA NRO: 36                          |
                                           |
CLASSE: APLICACAO                          |
CONCLUSAO: Terminal de video e'            |
           do tipo grafico                 |
CONECTIVO:                                 |
                                           |
                                           |
                                           |
                                           |
                                           |
                                           |
                                           |
==========================================================================
```

Os tipos de conectivos são:

[o] : conjunção;

[e] : disjunção;

[n] : negação;

[x] : ou-exclusivo;

Escolhido o conectivo, EDICON entra na fase de inserção dos fatos que farão parte das <CONDIÇÕES>. Neste instante, o sistema solicita ao EC qual o número da Classe da primeira evidência a fazer parte das <CONDIÇÕES>.

```
==========================================================================
OPERAR : <num>SELECAO, [i]NSERIR, [e]LIMINAR, [m]OVER, [a]LTERAR
CLASSES  [Esc]SAIR:
==========================================================================
    REGRA NRO: 36                          | AS CLASSES ATUAIS SAO
                                           |
CLASSE: APLICACAO                          | 1.APLICACAO
CONCLUSAO: Terminal de video e'            | 2.HARDWARE
           do tipo grafico                 | 3.CUSTOS
CONECTIVO: &                               | 4.SOFTWARE
CONDICOES:                                 |
                                           |
                                           |
                                           |
                                           |
                                           |
                                           |
==========================================================================
```

Escolhida a Classe, EDICON apresenta a lista de qualificadores associados a esta classe. O EC deverá escolher um qualificador e seguidamente um valor. Terminada a inserção de uma <CONDIÇÃO> temos:

```
==========================================================================
OPERAR : <num>SELECAO, [i]NSERIR, [e]LIMINAR, [m]OVER, [a]LTERAR
VALORES  [Esc]SAIR:
==========================================================================
    REGRA NRO: 36                          | QUALIFICADOR:
                                           |  O destino aplicacoes e'
                                           |
CLASSE: APLICACAO                          |
CONCLUSAO: Terminal de video e'            | 1.producao de relatorios
           do tipo grafico                 | 2.gerenciamento de B.D.
CONECTIVO: &                               | 3.processamento de texto
CONDICOES:                                 | 4.processamento grafico
  CLASSE: APLICACAO                        | 5.telecomunicacao
    1. O destino das aplicacoes e'         | 6.redes locais
       desenvolver software                | 7.educacao
                                           | 9.divertimento
                                           |10.desenvolver software
                                           |11.sistema especialista
                                           |
                                           |
==========================================================================
```

O EC pode, se desejar, inserir uma nova <CONDIÇÃO> com o mesmo Qualificador e a mesma Classe, bastando apenas, escolher um novo valor. O sistema apresenta, automaticamente, uma nova <CONDIÇÃO>.

```
==========================================================================
OPERAR : <num>SELECAO, [i]NSERIR, [e]LIMINAR, [m]OVER, [a]LTERAR
VALORES  [Esc]SAIR:
==========================================================================
    REGRA NRO: 36                          | QUALIFICADOR:
                                           |  O destino aplicacoes e'
CLASSE: APLICACAO                          |
CONCLUSAO: Terminal de video e'            | 1.producao de relatorios
           do tipo grafico                 | 2.gerenciamento de B.D.
CONECTIVO: &                               | 3.processamento de texto
CONDICOES:                                 | 4.processamento grafico
  CLASSE: APLICACAO                        | 5.telecomunicacao
    1. O destino das aplicacoes e'         | 6.redes locais
       desenvolver software                | 7.educacao
    2. O destino das aplicacoes e'         | 9.divertimento
       sistema especialista                |10.desenvolver software
                                           |11.sistema especialista
                                           |
                                           |
==========================================================================
```

Terminada a inserçao das <CONDIÇÕES> com mesmo Qualificador, o EC retorna ao nível anterior, entrando com a opção "Esc", e começa a inserir as <CONDIÇÕES> com mesma Classe.

O EC após inserir todas as evidências da mesma classe, retorna ao nível anterior, onde irá escolher, se necessário, uma nova Classe para inserir novas <CONDIÇÕES>. Este procedimento se repetirá até o EC completar o número de <CONDIÇÕES> desejadas.

A seguir, EDICON pergunta qual o FATOR DE ATENUAÇÃO da regra e o usuário deverá entrar com um valor entre 0.1 a 1.0. neste instante temos:

```
==========================================================================
QUAL E' O FATOR DE ATENUACAO (0.01 a 1.00)?

==========================================================================
    REGRA NRO: 36                          |
                                           | Software ser desenvdo. e'
CLASSE: APLICACAO                          |
CONCLUSAO: Terminal de video e'            | 1.para uso grafico
           do tipo grafico                 | 2.para uso nao grafico
CONECTIVO: &                               |
CONDICOES:                                 |
  CLASSE: APLICACAO                        |
    1. O destino das aplicacoes e'         |
       desenvolver software                |
    2. O destino das aplicacoes e'         |
       sistema especialista                |
  CLASSE: SOFTWARE                         |
    3. Software a ser desenvolvido e'      |
       para uso grafico                    |
                                           |
                                           |
==========================================================================
```

Terminada a edição de uma regra, e utilizando o conceito de hierarquização de regras, o EDICON pergunta ao EC se deseja inserir outra regra utilizando a mesma <CONCLUSÃO>. Isso é feito, apresentando o número da próxima regra e a mesma <CONCLUSÃO> da regra anterior, e perguntando o novo conectivo.

```
==========================================================================
QUAL O CONECTIVO A SER USADO NA REGRA?
[o]ou, [e]&, [n]nao, [x]ou-exclusivo, [Esc]sair:
==========================================================================
    REGRA NRO: 37                          |
                                           |
CLASSE: APLICACAO                          |
CONCLUSAO: Terminal de video e'            |
           do tipo grafico                 |
CONECTIVO:                                 |
                                           |
                                           |
                                           |
                                           |
                                           |
                                           |
==========================================================================
```

Caso o EC escolha um conectivo, ele deverá fornecer as novas <CONDIÇÕES>. Caso não deseje continuar, ele sai desse nível, voltando ao nível anterior. No nível anterior, o EC é indagado se deseja inserir uma nova regra utilizando o mesmo qualificador na <CONCLUSÃO>.

```
==========================================================================
OPERAR : <num>SELECAO, [i]NSERIR, [e]LIMINAR, [m]OVER, [a]LTERAR
VALORES  [Esc]SAIR:
==========================================================================
    REGRA NRO: 37                        | QUALIFICADOR:
                                         |  O terminal de video e'
CLASSE: APLICACAO                        |
CONCLUSAO: Terminal de video e'          |  1.do tipo texto
                                         |  2.do tipo grafico
                                         |
                                         |
                                         |
                                         |
                                         |
                                         |
                                         |
                                         |
                                         |
==========================================================================
```

Da mesma forma, quando o EC sai desse nível, poderá escrever uma regra aproveitando a mesma classe. Note que dessa forma, as regras sempre estarão agrupadas, o que ajuda o EC a não repetir regras e a organizar estruturadamente a sua BC.

O EDICON ao voltar ao primeiro nível, ou seja, à tela de OPERAR REGRAS, apresenta a parte <CONCLUSÃO> de todas as regras na ordem de inserção. O EC neste momento, poderá inspecionar e corrigir sua BC. Quando ele estiver satisfeito, entrará com a opção "Esc" para sair do editor e voltar a BACKFORNEB.

Neste momento, estará terminada a ediçao da BC. O usuário, voltando a BACKFORNEB, poderá gravar, executar e/ou listar essa BC, a fim de testá-la e corrigí-la.

3.2 UTILIZAÇÃO DO MOTOR DE INFERÊNCIAS

O Sistema Diagnosticador BACKFORNEB é utilizado para determinar as hipóteses mais promissoras, à partir de um conjunto de fatos (também chamados de evidências). Cada fato tem associado um GC, que varia entre GCMin (negação absoluta) a GCmax (afirmação absoluta). O valor (GCmax + GCmin)/2 será usado para informar sobre fatos que não foram observados ou que não são possíveis de se responder imediamente (*).

BACKFORNEB auxilia na análise dos fatos porque possui um alto nível de conhecimento, faz a avaliação com muito mais rapidez e precisão, e é capaz de "lembrar" todas as perguntas necessárias.

O Sistema Diagnosticador BACKFORNEB objetiva auxiliar o usuário a determinar, em um tempo relativamente curto, qual ou quais são as hipóteses mais prováveis para um específico problema, a partir do conhecimento dos fatos observados.

(*) Os exemplos a seguir , bem com na atual implementação de BACKFORNEB, serão utilizados os valores GCmin = -5, GCmax = 5 e 0 para valores não observados.

Para o sistema chegar a um diagnóstico, ele comporta-se da seguinte maneira:

O usuário inicialmente fornece as evidências observadas, de forma voluntária ("fase de voluntariamento"). De posse dessas informações, o sistema fará uma análise em "forward channing" (encadeamento para frente: Fatos -> Hipótese) e apresentará as hipóteses mais promissoras, em ordem decrescente de promissoriedade.

Por exemplo, após a fase de voluntariamento, teremos os seguintes resultados:

HIPÓTESE	PROMISSORIEDADE	
1. O equipamento necessário é IBM PC-XT	5.00	(100%)
2. Driver necessário é para disco de face dupla	4.00	(90%)
3. A impressora é qualidade carta	-2.50	(25%)

Após apresentar a lista das prováveis hipóteses, o usuário escolhe uma hipótese da lista e o Sistema Diagnosticador tenta prová-la em "backward channing" (encadeamento para trás: Hipótese -> Fatos). Durante a prova, as perguntas que se fizerem necessárias (sobre evidências específicas), são feitas ao usuário. A cada resposta do usuário, ou seja, a confirmação (ou não) de uma evidência, o GC desse fato será propagado (em "forward channing") em direção a hipótese.

BACKFORNEB termina a consulta quando tiver esgotado todas as perguntas, ou quando o usuário quiser suspender a consulta para retomada posterior.

3.2.1 Funcionamento do Sistema

O Sistema Diagnosticador BACKFORNEB apresenta uma série de mensagens durante a consulta do usuário. Ao iniciar uma sessão, BACKFORNEB mostra a seguinte tela:

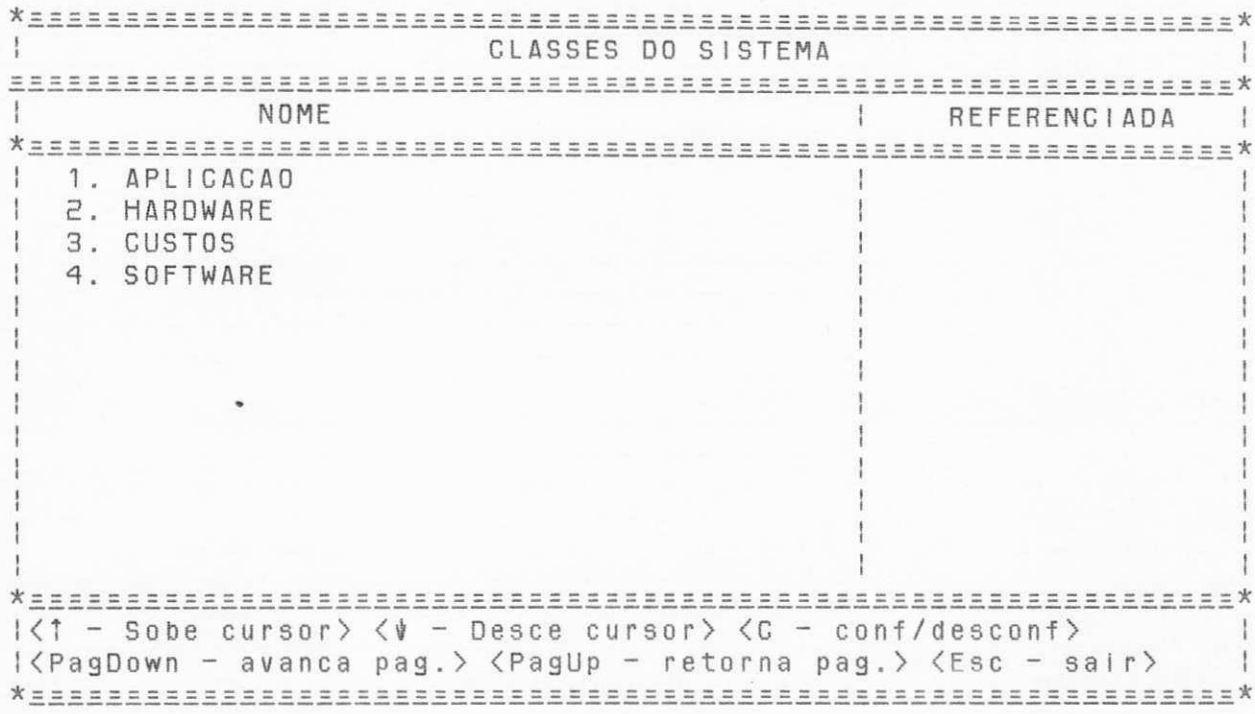
```
BEM VINDO AO SISTEMA DIAGNOSTICADOR BACKFORNEB
==============================================================================

S: CODIGO DO DIAGNOSTICANDO (MAX 8 CARACTERES)?
   [Esc]sair, [e]liminar diagnosticando
U:

    DIAGNOSTICADORES QUE USARAM O SISTEMA

     1. MCRAMOS
     2. PTGALUCI
     3. VLSOUZA
      .
      .
    15. MFCAMPOS
    16. ACSANTOS

==============================================================================
```

O usuário entra com o código do Diagnosticando e a seguir, o sistema verifica se o Diagnosticando é novo para ele. Caso isso ocorra, faz-se uma série de perguntas sobre suas características básicas e seus dados invariantes que serão guardados em um arquivo padrão ("Dgdor.TXT"). No caso de consulta médica de um paciente, esse arquivo seria denominado Anamnase.

```
SISTEMA DIAGNOSTICADOR BACKFORNEB
=================================================================

DIAGNOSTICANDO ADIMITIDO POR BACKFORNEB
S: DIAGNOSTICANDO E' NOVO
   INFORME CARACTERISTICAS BASICAS

   NOME: Maria Cristina Ramaos
   ENDEREÇO: Rua da Penha, 32
   CIDADE: C. Grande - Pb
   IDADE: 35
   SEXO (masc/fem): fem


=================================================================
```

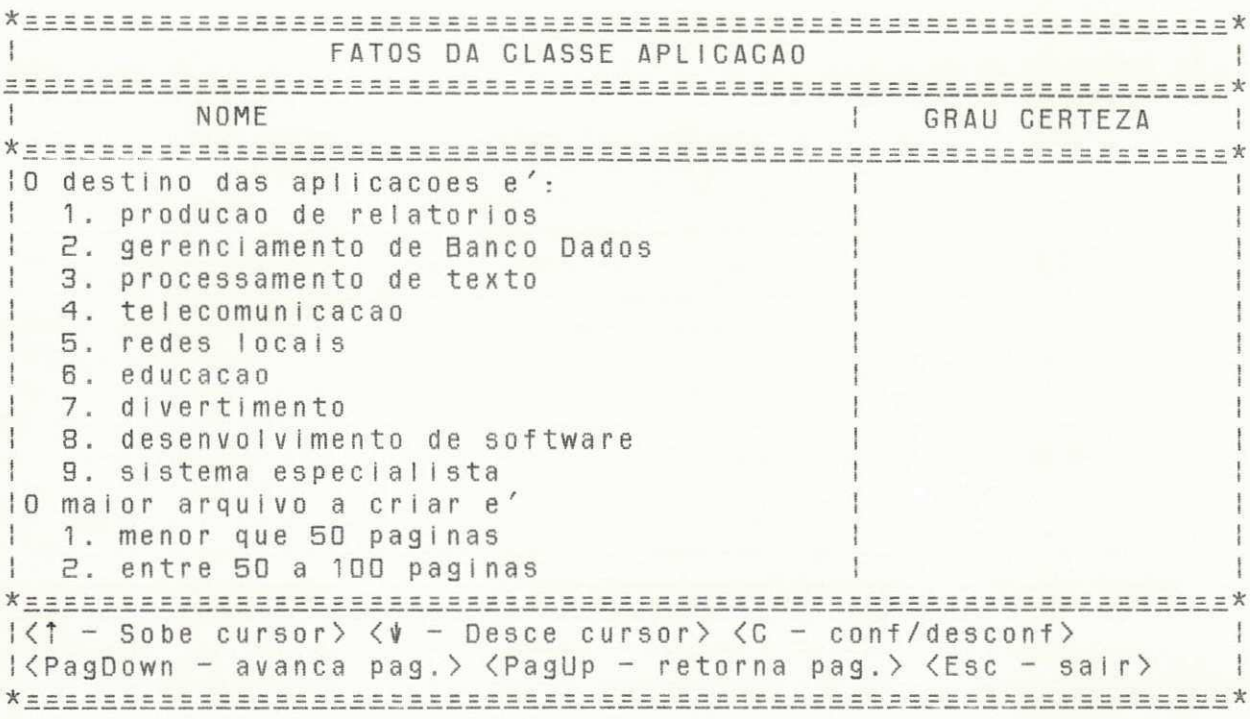
Se isto não ocorrer é porque o Diagnosticando já consultou o sistema e portanto existe o arquivo de dados invariantes e também o arquivo contendo o estado atual da consulta. BACKFORNEB pergunta ao usuário se ele deseja continuar a consulta que estava suspensa. Caso o usuário diga sim, o sistema traz para a memória esses arquivos e a partir daí, passa a perguntar os fatos que ficaram suspensos na consulta anterior. Caso contrário, se o Diagnosticando for velho mas deseja iniciar uma nova consulta, o sistema carrega apenas o arquivo de dados invariantes.

Após esta etapa, o Sistema Diagnosticador passa para a fase de voluntariamento dos fatos, onde o usuário irá escolher dentre todas as possíveis evidências, aquelas que são mais facilmente observadas. Isto é feito apresentando-lhe um menu das classes de evidências.



O usuário pode mover o cursor para cima (" ↑ ") ou para baixo (" ↓ ") fixando-o na Classe desejada. Pode avançar para a próxima pagina ("PagDown") e o sistema irá apresentar as Classes restantes que não couberam na tela atual. O mesmo ocorrerá caso o usuário desejar retornar à pagina anterior ("PagUp"), ou seja, listar as regras anteriores.

Confirmando-se uma ou mais classes, BACKFORNEB apresenta um novo menu com os Qualificadores do grupo escolhido e seus respectivos valores para que sejam assinalados os desejados. Por exemplo, se a classe APLICAÇÃO foi confirmada (escolhida), o sistema apresentará a seguinte tela:



O menu de opções dos FATOS DA CLASSE APLICAÇÃO é igual ao de CLASSES DO SISTEMA. Quando o usuário confirmar um valor, o cursor irá se deslocar para o campo do Grau Certeza e aguardará que o usuário forneça o respectivo GC. Se caso for a ativação de uma consulta antiga, serão apresentados os respectivos GC's dos fatos que foram voluntariados anteriormente. Neste momento, o usuário poderá modificar o valor do GC, ou simplesmente anulá-lo, bastando que em vez de fornecer um novo GC, ele entre com <return>. Cabe salientar que serão apresentados somente as Classes / Qualificadores / Valores que podem ser voluntariados, isto é, as folhas da Rede de Inferências.

Após assinalar todas as evidências observadas, o usuário encerra a etapa de voluntariamento entrando com a opção "Esc" dentro do menu de classes. Neste momento BACKFORNEB, que já propagou essas evidências através da sua BC, apresenta as

hipóteses em ordem decrescente de promissoriedade. Temos então, por exemplo, uma tela com o seguinte conteúdo:

```
SISTEMA DIAGNOSTICADOR BACKFORNEB
=================================================================

S: HIPOTESES MAIS PROMISSORAS (ORDEM DECRESCENTE), ESCOLHA UMA:
     1. O equipamento necessario e' IBM PC-XT.............4.00
     2. A impressora e' qualidade carta ..................4.00
     3. Driver necessario e' para disco de face dupla .....3.00
     4. Placa necessaria e' para display colorido/grafico..2.00
     <ENTER> - LISTA AS HIPOTESES NOVAMENTE
       <v>   - VOLUNTARIAMENTO DE FATOS
      <Esc>  - ABANDONE DIAGNOSTICADOR
U:

=================================================================
```

O usuário terá então que escolher qual a hipótese que ele quer que seja avaliada. Note que não é necessário que se escolha aquela com maior promissoridade, pode-se querer avaliar uma hipótese que pareça mais urgente ou mais importante no momento.

Durante a consulta, o Diagnosticando pode lembrar de algum(s) fato(s) e querer voluntariá-lo(s). Para isso, BACKFORNEB tem a opção de, a qualquer momento, o usuário poder iniciar o voluntariamento de evidências, digitando a opção "v". Nesse instante, BACKFORNEB apresenta o menu das classes de evidências e fica aguardando as escolhas do usuário.

Ao terminar o voluntariamento, o sistema apresenta a lista de hipóteses mais promissoras com suas respectivas promissoriedades recalculadas em função das evidências assinaladas.

O usuário poderá então escolher uma hipótese, e BACKFORNEB entrará na fase de perguntas, onde questionará ao usuário as evidências necessárias para se provar a hipótese escolhida. As evidências assinaladas na fase de voluntariamento não serão mais perguntadas. Se por exemplo, o usuário escolher a hipótese número 2, teremos:

```
SISTEMA DIAGNOSTICADOR BACKFORNEB
=================================================================

HIPOTESE SENDO PROVADA: A impressora e' qualidade carta

OPÇÕES: <-5> a <5> - valor do Grau de Certeza (GC)
          <ENTER> - fato nao observado
            <v>   - voluntariamento
            <s>   - sair da fase de perguntas

S: O custo e' fator importante?
U: 4
S: Tipo carta com maior custo e' nao essencial?
U: v
```

===

Durante a fase de perguntas o usuário também poderá voluntariar fatos, bastando digitar a opção "v". Ao retornar do voluntariamento, o sistema volta a fazer a pergunta que estava

suspensa, e continua a consulta. Terminada todas as perguntas (por exaustão ou por interrupção do usuário), BACKFORNEB apresenta novamente as hipóteses mais promissoras, com suas respectivas promissoriedades já recalculadas. Após ter respondido às perguntas do sistema ou ter voluntariado novos fatos, poderiamos ter a seguinte tela:

```
SISTEMA DIAGNOSTICADOR BACKFORNEB
=================================================================

S: HIPÓTESES MAIS PROMISSORAS (ORDEM DECRESCENTE), ESCOLHA UMA:
  1. O equipamento necessario e' IBM PC-XT ..... 5.00
  2. Driver necessario e' para disco de face dupla ..... 4.00
  3. A impressora e' qualidade carta ..... 3.50
  4. Placa necessaria e' para display colorido/grafico ... 2.00
  <ENTER> - LISTA HIPOTESES NOVAMENTE
    <v>   - VOLUNTARIAMENTO DE FATOS
   <Esc>  - ABANDONE DIAGNOSTICADOR
U:

=================================================================
```

Note que, neste caso, tivemos uma mudança na promissoriedade de algumas hipóteses. A consulta ao sistema terminará quando o usuário entrar com a opção Abandone Diagnosticador ("Esc"). Nesse momento será indagado ao usuário se ele deseja gravar os resultados obtidos na consulta. Se sim, o sistema armazena o nome do Diagnosticando no Arquivo de Diagnosticandos ('Dgando.Nom') e grava os resultados para uso posterior, no arquivo 'Dgando.GC'.

Terminada a sessão para um determinado Diagnosticando, BACKFORNEB pergunta se o usuário deseja reiniciar a consulta com novo Diagnosticando.

```
BEM VINDO AO SISTEMA DIAGNOSTICADOR BACKFORNEB
==============================================================================

S: CODIGO DO DIAGNOSTICANDO (MAX 8 CARACTERES)?
   [Esc]sair, [e]liminar Diagnosticando
U:

    DIAGNOSTICADORES QUE USARAM O SISTEMA

    1. MCRAMOS
    2. PTGALUCI
    3. VLSOUZA
    .
    .
    .
    15. MFCAMPOS
    16. ACSANTOS

==============================================================================
```

O usuário desejando encerrar a utilização do sistema, digitará "Esc" à pergunta "Nome do Diagnosticando" e BACKFORNEB apresentará a seguinte tela de encerramento:

```
SISTEMA DIAGNOSTICADOR BACKFORNEB
==============================================================================


   *** BACKFORNEB AGRADECE A SUA ATENCAO ***


                          ***  OBRIGADO  ***


==============================================================================
```

4. ASPECTOS DE IMPLEMENTAÇÃO DO SISTEMA BACKFORNEB

BACKFORNEB foi implementado na linguagem PASCAL. Na fase inicial do projeto, só possuíamos linguagens pouco convenientes para construção de software básico (FORTRAN, COBOL, BASIC, etc). PROLOG só dispunhamos de uma versão interpretada, sem acesso direto a arrays, e portanto lenta. C e PASCAL (com leves vantagens para C), ambas as linguagens são reconhecidas como favorecendo modularidade, estruturação , portabilidade, eficiência de execução, produtividade do programador, etc., sendo ótimas opções para software básico. Descartamos C apenas porque não compensaria investirmos meses para adiquirirmos nela a fluência que já tinhamos em PASCAL.

4.1 ESTRUTURA DE DADOS

As regras da Base de Conhecimento estão organizadas em 4 tabelas que são:

- Tabela de Regras e Fatos (TabRegFat)
- Tabela de Classes/Qualificadores/Valores (TabCqv)
- Tabela de Filhos e Pais (TabFP)
- Tabela do número de Regras (Regras)

4.1.1 Tabela de Classes/Qualificadores/Valores (TabCqv)

A base de conhecimento está dividida em várias Classes, onde cada Classe possui uma lista de Qualificadores e cada

Qualificador uma lista dos seus respectivos Valores. Por terem essas listas basicamente as mesmas características, foi utilizada uma única tabela (TabCqv) para representá-las, conforme Figura 4.1:

Nome	Utilizado	ProxElem	InicList
string[30]	int.	int.	int.

Figura 4.1 - Estrutura dos elementos de TabCqv

Cada elemento possui 4 campos, onde:

Nome - é o nome da Classe ou Qualificador ou Valor;

ProxElem - indica a posição (em TabCqv) do próximo elemento da lista;

InicList - é a posição do início da lista. Se for uma Classe, indica o início da lista dos respectivos Qualificadores. Caso seja um Qualificador, será o início da lista dos Valores a ele associados. O campo InicList está inativo para os Valores.

Utilizado- representa o número de regras que utilizam este registro (isto é importante para evitar que seja eliminado uma Classe/Qualif/Valor que está sendo usado em alguma regra). Se o elemento for um Valor, este campo indica a posição da regra que o está utilizando (a posição da regra é importante para a fase de Voluntariamento de Fatos).

4.1.2 Tabela de Regras e Fatos (TabRegFat)

Cada elemento da tabela é formado por 7 campos conforme é mostrado na Figura 4.2:

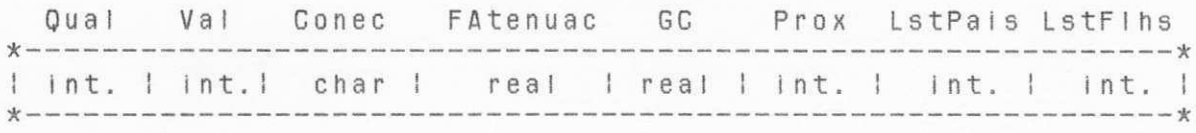


Figura 4.2 - Estrutura dos elementos de TabRegFat

Cada campo do elemento tem o seguinte significado:

Qual - posição em TabCqv do nome do qualificador;

Val - posição em TabCqv do nome do valor;

Conec - tipo de conectivo que pode ser "and", "or", "not", "ind", "xor" (só utilizado na Conclusão);

FAtenuac - fator de atenuação da regra (só utilizado na Conclusão);

GC - inicialmente, todos os GC's possuem o valor GCinic (grau de certeza inicial);

Prox - contém o endereço em TabRegFat do próximo registro que possui mesma chave (isto ocorre quando há colisão);

LstPais - possui o endereço em TabFP do início da lista dos seus pais;

LstFlhs - possui o endereço em TabFP do início da lista dos seus filhos;

Nesta tabela está armazenada toda a rede de inferências, ou seja, todos os nodos (<CONCLUSÃO>) com seus respectivos filhos (<CONDIÇÕES>). Esta tabela é o esqueleto

principal da representação do conhecimento.

Para um melhor aproveitamento de espaço, e melhor gerenciamento das estruturas de dados, visto que, por ser um processo dinâmico (pode ocorrer muitas inserções e eliminações de registros), optou-se por uma estrutura de lista encadeada em TabFP e TabCqv.

Em TabRegFat tornaria-se ineficiente a utilização de uma lista encadeada, pois devido a seu tamanho, o "caminhamento" nessa lista seria muito demorado, bem como a eliminação de uma regra. Neste caso, optou-se então pela utilização de uma função Hash (*) que através de uma chave (Qual + Valor), fornece a posição na tabela. Caso haja uma colisão (ou seja, a posição fornecida pela função Hash já estiver ocupada), o novo registro será inserido em uma lista de posições livres de TabRegFat. As colisões de uma mesma chave são "amarradas" através do campo PROX de cada registro.

Veremos na sessão 4.1.5 um pequeno exemplo de uma Base de Conhecimento e como está organizada sua estrutura de dados.

4.1.3 Tabela de Filhos e Pais (TabFP)

Esta tabela possui a lista dos filhos e dos pais de cada registro de TabRegFat (ou seja, de cada nodo da rede de inferências). Vide Figura 4.3:

(*) A funçao Hash utilizada é a do tipo [(A MOD B) + 1] onde A é a soma dos valores de QUAL e VALOR de TabRegFat menos 5 (para ajuste no deslocamento) e B será o tamanho da tabela (4*j+3) [HORW 77].

Elem	Prox
int.	int.

Figura 4.3 - Estrutura dos elementos de TabFP

O campo denominado Elem é utilizado para indicar a posição do registro em TabRegFat. O campo Prox determina o próximo elemento da lista.

4.1.4 Tabela do número de Regras (Regras)

Esta tabela contém, de forma sequencial, todos os elementos de TabRegFat que são regras (<CONCLUSÃO>). Esta tabela é importante para podermos apresentar ao usuário as regras da BC numeradas, de forma organizada e estruturada (Figura 4.4).

Elem	NroClass
int.	int.

Figura 4.4 - Estrutura dos elementos de Regras

Cada registro da Tabela de Regras possui 2 campos com o seguinte significado:

Elem - posição em TabRegFat da <CONCLUSÃO>;

NroClass - número da classe a que pertence o elemento.

4.1.5 EXEMPLO DA ORGANIZAÇÃO DAS ESTRUTURAS DE DADOS

BACKFORNEB utiliza como parâmetros os seguintes tamanhos para as estruturas de dados:

- MaxRegFat = 800 : tamanho da Tabela de Regras e Fatos;
- MaxCqv = 800 : tamanho da Tabela de Classes/Qualif/Valores;
- MaxFP = 1800 : número máximo de Filhos e Pais;
- MaxRegras = 500 : número máximo de Regras;
- TamTabPrinc = 503: número de posições utilizadas pela função Hash;
- (MaxRegFat - TamTabPrinc) = 297: tamanho da lista de posições livres de TabRegFat, utilizada quando houver colisões.

Serão utilizados para exemplificar o uso das tabelas, as seguintes regras:

REGRA NRO: 1

SE:

O destino das aplicacoes e' desenvolvimento de software

&

O software a ser desenvolvido e' para uso grafico

ENTAO:

Terminal de video e' do tipo grafico

FATOR DE ATENUACAO = 0.90

REGRA NRO: 2

SE:

Um grande espaco em disco e' necessario

&

Os arquivos a criar devem ser usados de uma vez

&

O custo e' fator importante

ENTAO:

O equipamento necessario e' IBM PC-XT

FATOR DE ATENUACAO = 0.80

As Figuras 4.5, 4.6, 4.7 e 4.8 apresentam a organização dos dados nas tabelas. Neste exemplo, o registro número 16 de TabRegFat diz que a primeira regra tem Qualificador = "Terminal de video e'" (posição 9 de TabCqv) e Valor = "do tipo grafico" (posição 11 de TabCqv); não tem pais pois é uma Conclusão; e a lista dos seus filhos se inicia na posição 1 de TabFP. Em TabFP conclui-se que os filhos (Condições) da regra 16 são os registros das posições 2 e 9 de TabRegFat, ou seja, "O destino das aplicacoes e' desenvolvimento de software" e "O software a ser desenvolvido e' para uso grafico".

Pos	Qual	Val	Conec	FAtenuac	GC	Prox	LstPais	LstFlhs
2	2	4	-	-	-	0	2	0
9	6	7	-	-	-	0	4	0
16	9	11	e	0.90	-	0	0	1
22	12	14	-	-	-	0	8	0
27	15	16	-	-	-	0	6	0
35	19	20	-	-	-	0	10	0
44	23	25	-	0.80	-	0	0	5

Figura 4.5 - Organização dos dados em TabRegFat

Pos	Elem	Prox
1	16	1
2	44	22

Figura 4.6 - Organização dos dados em Regras

Pos	Nome	Utilizado	ProxElem	InicList
1	APLICACAO	5	18	2
2	O destino das aplicacoes e'	1	6	3
3	educacao	0	4	0
4	desenvolvimento de software	2	5	0
5	sistemas especialistas	0	0	0
6	Software a ser desenvolv. e'	1	9	7
7	para uso grafico	9	8	0
8	para uso nao grafico	0	0	0
9	O terminal de video e'	1	12	10
10	do tipo texto	0	11	0
11	do tipo grafico	16	0	0
12	Os arquivos a criar	1	15	13
13	podem ser seccionados(<100p)	0	14	0
14	devem ser usados de uma vez	22	0	0
15	Um grande espaco em disco e'	1	0	16
16	necessario	27	17	0
17	nao necessario	0	0	0
18	Custos	1	22	19
19	O custo e'	1	0	20
20	fator principal	35	21	0
21	menos importante q/ flexib/e	0	0	0
22	Hardware	1	0	23
23	O equipamento necessario e'	1	0	24
24	IBM PC	0	25	0
25	IBM PC-XT	44	0	0

Figura 4.7 - Organização dos dados em TabCqv

Pos	Elem	Prox
1	2	3
2	16	0
3	9	0
4	16	0
5	27	7
6	44	0
7	22	9
8	44	0
9	35	0
10	44	0

Figura 4.8 - Organização dos dados em TabFP

4.2 IMPLEMENTAÇÃO DO MOTOR DE INFERÊNCIAS

O funcionamento básico de BACKFORNEB com seus módulos principais, será mostrado agora em forma de algorítmo (em alto nível).

a) **Módulo Principal**

O módulo principal implementa o gerenciador do sistema

onde, após obter o nome do Diagnosticador (Dgdor) e o nome do Dgando, irá carregar a BC do Dgdor e as informações do Dgando caso existam. Após esta etapa, BACKFORNEB passa para a fase de voluntariamento de fatos. Após obter todos os fatos voluntariados e consequentemente propagado pela rede os seus respectivos GC's, o sistema apresenta as hipóteses mais promissoras. Escolhida uma hipótese, o sistema irá fazer , em "backward channing", as perguntas necessárias para provar ou refutar a hipótese. Durante a prova, o usuário pode fazer o voluntariamento/correções de fatos que desejar. Terminada a prova, e encerrada a consulta pelo usuário, BACKFORNEB irá gravar a consulta do Dgando no arquivo "Dgando.GC", e alterar também, na tabela de ArqDgando (que possui o nome de todos os Dgandos que utilizaram o sistema) o tipo de Dgando para "VELHO".

Terminada a consulta para um Dgando, o sistema volta a perguntar o nome de um novo Dgando e reinicia a consulta.

==> Algorítmo do Módulo Principal <==

```
início  /* módulo principal */
   Obtenha Dgdor;
   repita para sempre  /* ciclo externo */
      Obtenha Dgando;
      Se Dgando é FIM
        então interrompa
      fim se;
      se Dgando_é_Novo
        então Obtenha_DadosInvariantes;
      senão  /* Dgando é velho */
        Carrega_Arquivo_DadosInvariantes;
        se Reativa_Consulta_Suspença
          então Carrega_GC_Dgando
        fim se
      fim se
```

```
        repita para sempre /* ciclo interno */
           Obtenha_Fatos_Voluntariados;
           Exiba_Hipóteses_Promissoras;
           Obtenha_Hipótese_Escolhida;
           se Hipótese_Escolhida é para encerrar consulta
             então interrompa
           fim se;
           Faça_Perguntas_HipEscolhida
           se Resposta é Voluntariamento
             então Obtenha_Fatos_Voluntariados
           fim se;
           Avalia_Plausibilidade_HipEscolhida;
        fim repita;  /* ciclo interno */
        Grava no arquivo Dgando.GC os G.Certezas do Dgando;
     fim repita  /* ciclo externo */
  fim algorítmo.
```

b) **Voluntariamento de Fatos**

Durante a fase de voluntariamento o usuário pode, através de menus, assinalar os fatos observados no Dgando. O sistema apresenta o menu das classes de evidências, e fica aguardando as suas confirmações. Para cada classe assinalada, o sistema apresenta os qualificadores e seus respectivos valores. O usuário pode então fornecer o seu Grau de Certeza, pode modificá-lo (fornecendo um novo valor), ou pode também anulá-lo, bastando apenas entrar com <return> (valor nulo), onde for pedido o GC.

A cada fato assinalado, o seu GC é propagado por toda a rede de inferências. Com isso, o tempo de avaliação da promissoriedade fica diluído, tendo-se um considerável ganho em tempo de resposta para avaliar as hipóteses mais promissoras.

```
==> algorítmo de Voluntariamento de Fatos <==
  Inicio  /* Voluntariamento */
```

```
    repita para sempre
       Mostra_Classes;
       se Encerra_Voluntariamento
         então interrompa;
       Obtem_ListaClasses;
       se Terminou_Classes
         então interrompa;
       repita para sempre
          Obtenha_Fato;
          se Encerra_Fatos
            então interrompa;
          Obtenha_GCerteza
       fim repita;  /* fatos */
    fim repita;  /* Voluntariamento */
  fim algorítmo.
```

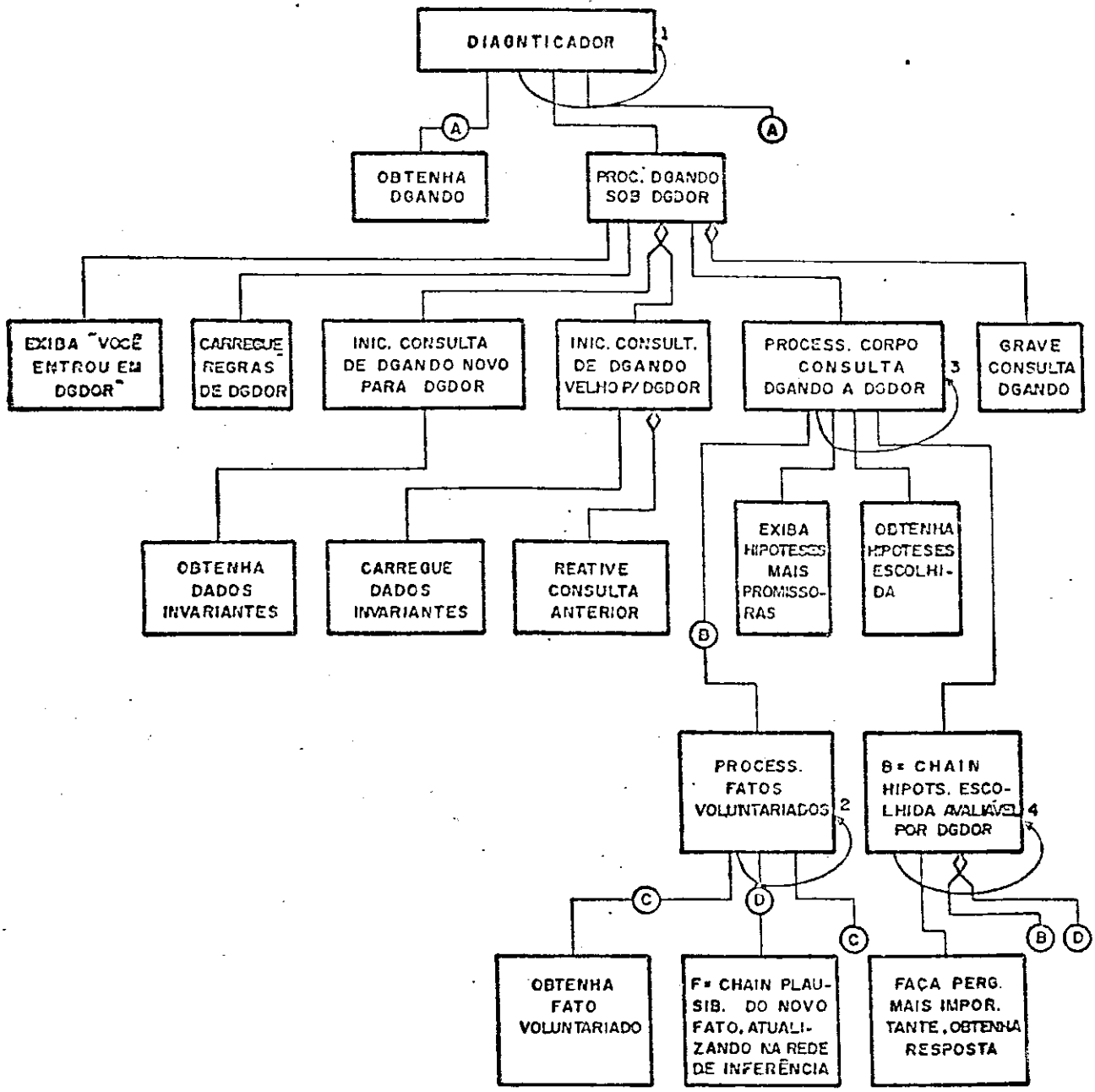
c) **Avaliação de Plausibilidade**

O módulo de Avaliação de Plausibilidade irá propagar o GC do fato para os seus pais da seguinte forma: inicialmente este módulo irá obter a lista dos pais do nodo e, para cada pai, irá atualizar o seu GC, levando em conta o GC do filho, o tipo de conectivo e o fator de atenuação (conforme visto em 2.4.1). Em seguida, o GC do pai será propagado, da mesma forma, em "forward chaining". A propagação terminará quando se atingir a raiz, e quando a lista dos pais do nodo terminar.

==> algorítmo de Atualiza Plausibilidade <==

```
  Atualiza_Plausibilidade (Nodo,GC);
    início
       se Nodo é a Raiz
         então interrompa
       fim se;
       Procura a lista dos pais do Nodo;
       enquanto Nodo tiver_pai faça
          Obtenha o GC do pai de acordo com o respectivo
                    conectivo e fator de atenuação;
          Atualiza o GC do pai;
          Atualiza_Plausibilidade (Pai)
       fim enquanto;
    fim algorítmo.
```

O gráfico estruturado do Motor de Inferências é apresentado na Figura 4.9:



1. ENQUANTO DGANDO ≠ "SAIR"
2. ENQUANTO TIVER FATOS VOLUNTARIADOS
3. ENQUANTO HIP. ESCOLHIDA ≠ "SIM"
4. ENQUANTO EXISTIR PERG. IMPORT. OU RESP= ENCERRE PERGUNTAS

Figura 4.9 - Gráfico Estruturado do Motor de Inferências

4.3 IMPLEMENTAÇÃO DO EDICON

O EDICON auxilia o EC a construir sua BC de forma organizada e estruturada. Tem facilidades para inserir, eliminar e modificar regras, e também faz com que o EC forneça, gradativamente, as regras que possuam a mesma conclusão, dentro da mesma classe e do mesmo qualificador. Este procedimento é importante para que as regras sejam editadas de forma agrupada. Na Figura 4.10 é apresentado o gráfico estruturado do EDICON.

4.4 ENTRADA/SAÍDA

BACKFORNEB possui 3 módulos de entrada/saída que são:

a) **Gravar BC**

A BC construída pelo EC será gravada em 3 arquivos. As estruturas das regras e fatos, isto é, os registros de TabRegFat, serão armazenados em um arquivo denominado "Dgdor.REG". As Classes / Qualificadores / Valores (TabCqv) serão gravadas no arquivo "Dgdor.CQV". As outras 2 tabelas que contém as listas dos filhos e pais (TabFP) e que contém a posição das regras (Regras), por possuirem a mesma estrutura, serão gravadas no arquivo "Dgdor.RFP".

Como normalmente as tabelas não são totalmente utilizadas, e após a edição de regras (inserção / deleção / modificação) suas posições estão desordenadas, então, para otimizar o tempo e espaço de armazenamento em disco desses

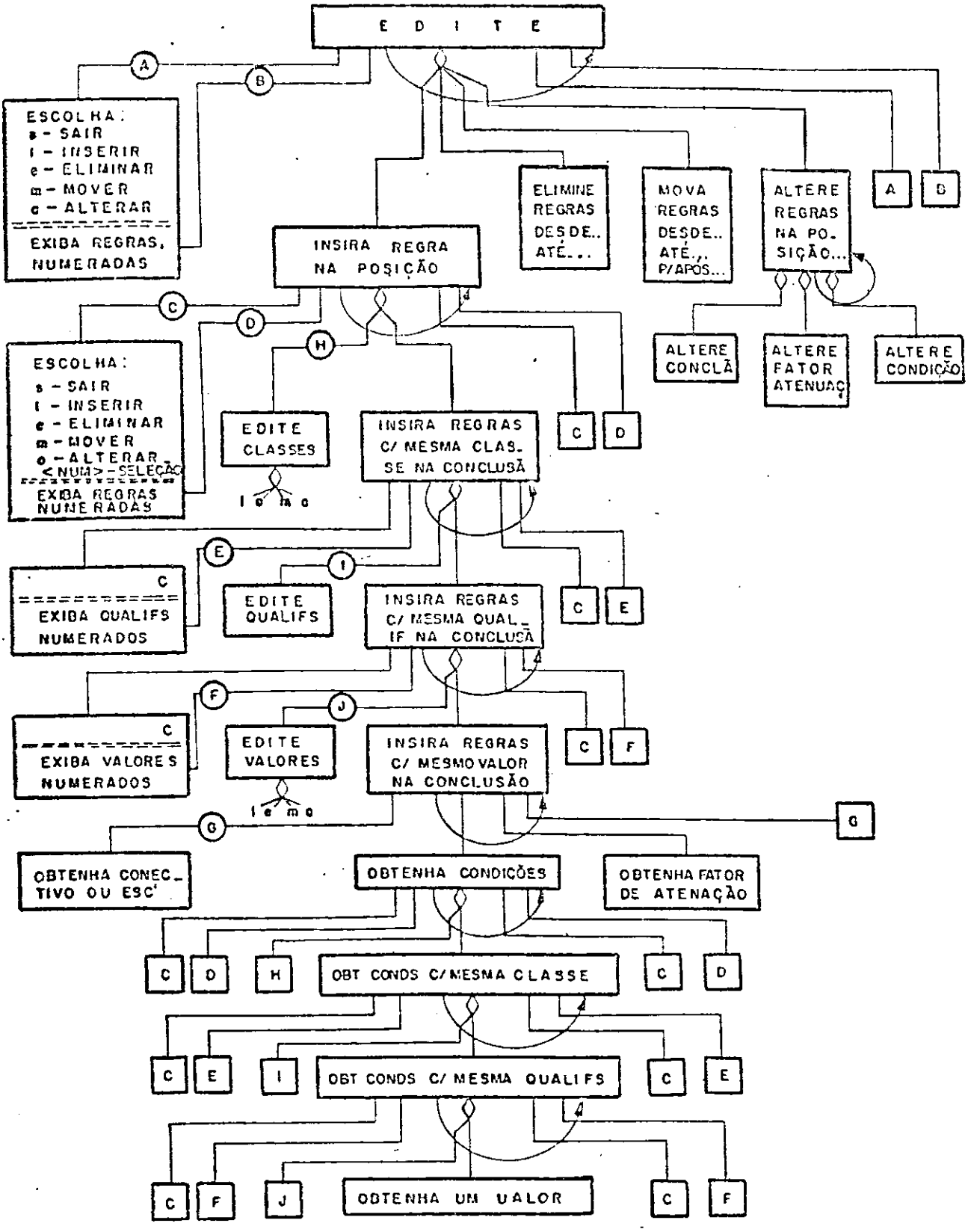


Figura 4.10 - Gráfico Estruturado do EDICON

registros, optou-se por gravar somente os registros ativos e, junto com cada um deles, a sua posição na tabela.

b) **Carregar BC**

Este módulo carrega o conhecimento armazenado em disco para as tabelas do sistema. O processo de restauração da BC existente é relativamente simples, visto que junto com cada registro lido do arquivo, é lido também a sua posição de origem.

c) **Imprimir Regras**

O EC periodicamente necessita de uma listagem (na impressora) das regras da BC que está editando. Portanto este módulo fornece todas as regras numeradas e também todas as Classes, Qualificadores e Valores existentes.

5. CONCLUSÕES E TRABALHOS FUTUROS

Embora o tempo não nos tenha permitido usar BACKFORNEB na construção de novos, ou reescrita de velhos SE's diagnosticadores de médio e grande porte, de modo a obter estatísticas e resultados bem significativos, usamos nossa ferramenta na reescrita de um pequeno SE (cerca de 105 regras) para configuração de sistemas computacionais tipo PC-IBM, que já havia sido escrito em EXSYS. Baseado nesta experiência, podemos dizer que:

a) BACKFORNEB apresentou-se bastante versátil e eficiente

b) Embora sendo um projeto de pesquisa acadêmica, esta ferramenta tem mostrado que possui um ótimo desempenho, pois sendo a avaliação de regras diluída durante a consulta, isto faz com que o seu tempo de resposta seja mínimo (em média de frações de segundo).

c) A nossa ferramenta utiliza 32 Kbytes de código objeto para o motor de inferências e mais 46 Kbytes para o editor, o que pode ser facilmente utilizada em microcomputadores.

d) Outras vantagens que podemos citar do nosso sistema é que BACKFORNEB:

- permite ao usuário voluntariar informações em qualquer ponto de uma consulta;

- tem suas inferências baseadas em encadeamento "forward" e "backward", com propagação dos graus de certeza através da rede de inferências;

- repetidamente fornece ao usuário uma lista das hipóteses mais promissoras naquele instante, e permite que ele escolha uma delas para ser perseguida. Além de dar ao usuário uma visão global dos resultados parciais já obtidos, isto permite que ele opte pela hipótese que lhe pareça mais urgente;

- é um sistema diagnosticador bastante flexível no que diz respeito à sua variedade de aplicações. Isto deve-se ao fato de que, diferentemente dos "shells", ele foi projetado sem influência de aplicações particulares.

Como sugestões para futuros trabalhos apresentamos:

1) Validação do sistema através da construção de um ou mais SE's reais, aproveitando toda a capacidade de BACKFORNEB para podermos melhor verificar sua completude, correção e eficiência.

2) Desenvolver um modo de explanação com justificativa ao usuário às perguntas tais como "explique o POR QUE da pergunta" e "exiba o COMO da prova" (com modalidade texto e gráfica).

3) Desenvolver um gerenciador de telas escrito em assembler, e também modo gráfico e colorido, para que tenhamos uma melhor interface homem-máquina.

4) Construir um módulo de aquisição de conhecimento que, além da edição pura e simples do conhecimento explicitamente relatado pelo EC, faça a checagem da consistência da BC, como também a indução / generalização / especialização de regras a partir do conhecimento já adquirido e também de novos exemplos.

5) implementar o conceito de sub-diagnosticadores conforme foi descrito em [AQUI 86], onde os Diagnosticadores seriam organizados de forma hierárquica e a consulta a um Dgando seria transferida, de forma recursiva, para o subdiagnosticador escolhido, e este faria exatamente o que o Dgdor anterior fez.

6) implementar varáveis simples e arrays (inteiras, reais, complexas, lógicas ou cadeias de caracteres), bem como expressões aritméticas e lógicas.

7) implementar interfaces com arquivos executáveis dotipo .COM, difundidos para gerenciamento de banco de dados (DBase, etc.) e de planilhas (Visicalc, etc.).

8) implementar um módulo acionador de testes automáticos, contra uma extensa bateria de casos-teste, a cada vez que uma regra for introduzida / modificada / eliminada.

9) implementar sofisticadas facilidades de tracing (acompanhamento) textual e gráfico e debugging (depuração).

10) permitir que o conteúdo dos dados invariantes sejam definidos pelo EC, diferentemente para cada Diagnosticador, sendo que esses dados devem influir na promissoriedade.

Estamos cientes que para tornar o nosso sistema mais flexível e mais poderoso, é necessário alguns pequenos trabalhos, dos quais citamos:

- executar qualquer hipótese em "backward channing" , mesmo que não tenha promissoriedade ou não haja voluntariamento;

- permitir ao EC a facilidade de escolher uma regra qualquer (independente de ser hipótese ou não) para ser perseguida (rovada). Isto auxiliaria na depura,ča de uma sub-´avore da rede;

- na eliminação indevida de uma classe/qualificador/valor, deverá haver, na mensagem de erro, quais são as regras em que ele é usado;

- dar ao usuário a possibilidade de definir o intervalo do GC, ou então, de trabalhar com Sim/Não em vez de GCmax/GCmin;

- incluir arquivos comentários (etiologia, prognóstico, etc.) para cada conclusão.

- incluir arquivos históricos (resumos de diagnósticos passados, tratamentos passados e seus resultados, etc.) para cada diagnosticando.

REFERÊNCIAS BIBLIOGRÁFICAS

[AQUI 86] AQUINO, M.S.; SILVA, H.M.; MONGIOVI, G. "Proposta do módulo de inferências para um sistema diagnosticador hierárquico de uso geral". III Simpósio Brasileiro de Inteligência Artificial, R. Janeiro, Novembro 1986.

[AQUI 87] AQUINO, M.S.; MONGIOVI, G.; SILVA, H. "EDICON - Editor de conhecimento para sistemas especialistas". IV Simpósio Brasileiro de Inteligência Artificial, Uberlândia, Outubro 1987.

[BARR 81] BARR, A.; FEIGENBAUN, E. "The handbook of artificial inteligence". Los Altos, William Kaufman, 1981, v.1.

[BUCH 78] BUCHANAN, B.G., FEIGENBAUM, E.A. "DENDRAL E META-DENDRAL: Their applications dimension", Artificial Intelligence 11, 1978.

[BUCH 84] BUCHANAN, B. G.; SHORTLIFFE, E.A. "Rule-based expert systems". Menlo Park, Addison-Wesley Publishing Company, 1984.

[CHIA 86] CHIANCA, M.P. "OFTALMO: Um sistema especialista para diagnóstico de síndromes oculares", dissertação de mestrado COPIN/UFPb, Campina Grande, 1986.

[DAVI 76] DAVIS, R. "Applications of meta level knowledge to the construction, maintenance and use of large knowledge bases". Ph.D thesis, Stanford, Computer Science Department, Stanford University, 1976.

[DUDA 81] DUDA, R.O.; GASCHINIG, J.G. "Knowledge-based expert systems come of age". Byte, Sept. 1981.

[ERMA 81] ERMAN, L.D.; LONDON, P.E.; FICKLAS, S.F. "The design and an example use of HEARSAY-III". In: INTERNATIONAL JOINT CONFERENCE ON ARTIFICIAL INTELLIGENCE, 7., Vancouver, Canada, 1981. Proceedings, Stanford, CA, Stanford University, 1981.

[EXSY 86] Manual do "EXSYS: Software para desenvolvimento de sistemas especialistas". Base Tecnologia Ltda., R. Janeiro, 1986.

[FAGA 80] FAGAN, J.M. "VM: Representing time-dependent relations in a medical setting". Doctoral Dissertation, Computer Science Department, Stanford University, Stanford, CA, June 1980.

[FAIN 82] FAIN, J.; HAYES-ROTH, F.; SOWIZRAL, H.;WATERMAN, D. "Programing in ROSIE: an introduction by means of examples". Santa Monica, CA, Rand, 1982.

[FORG 81] FORGY, C.L. "The OPS5 user's manual". Pittsburg, PA. Computer Science Department, Carnegie-Mellon University, 1981.

[FORS 84] FORSYTH, R. "Expert Systems". New York, Chapman and Hall, 1984.

[GREI 80] GREINER, R. "RLL-1: A representational language language". Stanford, CA, Heuristic Programming Project, Computer Science Department, Stanford University, 1980.

[HAYE 83] HAYES-ROTH, F.; WATERMAN, D.A.; LENAT, D.B. "Buiding expert systems". Addison--Wesley Publishing Company, Inc., v.1, 1983.

[HORW 77] HOROWITZ, E. "Fundamentals of data structures". Computer Science Press, Inc., 1977.

[McDE 80] McDERMOTT, J. "R1: A rule-based configurer of computer systems". Department of Computer Science, Carnegie-Mellon University, Rept. CMU-CS-80-119, April 1980.

[MELL 74] van MELLE, W. "Do you wish advice another horn?". Stanford Computer Science Department, Stanford University, 1974.

[NICO 87] NICOLLETTI, P.S. "SINDROMUS: Um sistema especialista para diagnóstico de síndromes de malformações congênitas". Dissertação de mestrado COPIN/UFPb, Campina Grande, 1987, e IV SBIA, Uberlândia - MG, 1987.

[NII 79] NII, H.P.; AIELLO, N. "AGE (attemp to generalize): A knowledge-based program for building knowledge-based programs". In: INTERNATIONAL JOINT CONFERENCE ON ARTIFICIAL INTELLIGENCE, 6., Tokyo, 1979, v.2.

[SAND 85] SANDRI, Sandra A. "Sistema Diagnóstico: uma ferramenta para construção de sistemas especialistas". S.J.Campos, INPE, 1985.

[SHOR 76] SHORTILIFFE, E. H. "Computer-Based Medical Consultations: MYCIN". Elsevier, New York, 1976.

[STEF 81] STEFIK, M. J. "Planning with constraints". Artificial Inteligence, 1981.

[STEF 82] STEFIK, M. et alii. "The organization of expert systems, a tutorial". Artificial Intelligence, 18, 1982.

[VPX 86] Manual do VP-Expert.

[WATE 85] WATERMAN, D. "A guide to expert systems". Menlo Park, Addison-Wesley Publishing Company, 1985.

[WEIS 79] WEISS, S.M.; KULIKOWSKI, C.A. "EXPERT: a system for developing consultation models". In: INTERNATIONAL JOINT CONFERENCE ON ARTIFICIAL INTELLIGENCE, 6., Tokyo, 1979. Proceedings. Stanford, CA, Stanford University, 1979, v.2.