Universidade Federal de Campina Grande
Centro de Ciências e Tecnologia
Coordenação de Pós-Graduação em Informática

Dissertação de Mestrado

Sistemas Especialistas como ferramenta auxiliar para o ensino da disciplina Bases da Técnica Cirúrgica.

João Fernandes Britto Aragão

Aracaju - Campina Grande
2002

Universidade Federal de Campina Grande
Centro de Ciências e Tecnologia
Coordenação de Pós-Graduação em Informática

João Fernandes Britto Aragão

Sistemas Especialistas como ferramenta auxiliar para o ensino da disciplina Bases da Técnica Cirúrgica.

Dissertação apresentada ao curso de Mestrado em Informática do Centro de Ciências e Tecnologia da Universidade Federal de Campina Grande, como requisito parcial para a obtenção do título de Mestre em Informática.

Área de concentração: Informática
Linha de pesquisa: Modelos Computacionais e Cognitivos

Orientador: Prof. Dr. Antônio Ramirez Hidalgo

UNIVERSIDADE FEDERAL DE CAMPINA GRANDE
Campina Grande
2002.

Ficha Catalográfica

Aragão, João Fernandes Britto.

A659S

Sistemas Especialistas como ferramenta auxiliar para o ensino da disciplina Bases da Técnica Cirúrgica.

Dissertação (Mestrado), Universidade Federal de Campina Grande, Centro de Ciências e Tecnologia, Coordenação de Pós-Graduação em Informática, Campina Grande - Pb, Julho de 2002.

220 páginas p. II

Orientador: Dr. Antonio Ramirez Higalgo

Palavras-Chave:

1. Inteligência Artificial
2. Sistemas Especialistas
3. Sistemas Cognitivos
4. Grafos
5. Distúrbios Hidroeletrolíticos
6. Nutrição em Cirurgia

CDU - 007.52

"SISTEMAS ESPECIALISTAS COMO FERRAMENTA AUXILIAR PARA O ENSINO DA DISCIPLINA BASES DA TÉCNICA CIRÚRGICA"

JOÃO FERNANDES BRITTO ARAGÃO

DISSERTAÇÃO APROVADA EM 01.07.2002

PROF. ANTONIO RAMIREZ HIDALGO, Dr.
Orientador

PROF. MARCUS COSTA SAMPAIO , Dr.
Examinador

PROF. JUGURTA ROSA MONTALVÃO FILHO, Dr.
Examinador

ARACAJU – SE

A Deus
Aos meus pais,
a minha esposa Régia e aos
meus tesouros:Priscila,Camila,Mariana e Nathália.

Agradecimentos

Minha mais sincera expressão de agradecimento aos meus amigos e mestres, os professores do DSC da UFCG, aos meus colegas da UNIT e da UFS pelo incentivo e estímulo para conclusão da presente dissertação.

Ao colega e amigo João Fonseca, por ter sido um bom aconselhador nas horas dos impasses.

A Ramirez, pela paciência e bons conselhos.

A minha Irmã Rosa Helena, agradeço sinceramente seu apoio.

Finalmente, agradeço a minha família pela compreensão e força dispensadas.

Glossário

Anamnese: Colheita de dados subjetivos sentidos pelos doentes e relatados ao médico.

Anastomose: Criação, através de suturas, de comunicação entre duas vísceras ocas.

Angiografia Digital: Estudo radiológico dos vasos sanguíneo com utilização de equipamentos que fazem uso de processamento digital de imagens.

Anorexia: Falta de apetite.

Anúrico: Sem urinar.

Apendicite: Inflamação do apêndice cecal.

Assistolia: Sem batimentos cardíacos.

Balanço Hidroeletrolítico: Mensuração da diferença entre a quantidade de líquidos e eletrólitos que são administrados ao paciente e eliminados pelo organismo. Quando predomina entradas o balanço é dito positivo. Quando as perdas são predominantes o balanço é dito negativo.

Bradicardia: Diminuição da freqüência cardíaca.

Cefaléia: Dor de cabeça.

CID: Código Internacional de Doenças. Atualmente na versão 10.

Colangite: Inflamação das vias biliares.

Choque: Situação caracterizada por diminuição do aporte de sangue para as células devido hipotensão arterial.

Desidratação hipertônica: Desidratação dom osmolaridade plasmática menor que 310 mOsm/l.

Desidratação hipotônica: Desidratação com osmolaridade plasmática menor que 210 mOsm/l.

Desidratação isotônica: Desidratação com osmolaridade normal (280 – 310 mOsm/l).

Desidratação: Distúrbio caracterizado pela diminuição da quantidade de água no organismo.

Desordens Reumatológicas: Alterações patológicas dos componentes das articulações.

Digital: Medicação utilizada para tratamento da Insuficiência Cardíaca.

Distúrbios Hidroeletrolíticos: Desordens provenientes de alterações absolutas ou relativas da água e eletrólitos (sódio, cloro, potássio, cálcio, magnésio, etc) em nosso organismo.

Distúrbios Metabólicos: Alterações decorrentes do aumento da concentração de hidrogênio ionte (Acidoses) ou da sua diminuição (Alcaloses) em nosso organismo.

Dispnéia: Dificuldade respiratória.

Diurese: Produção de urina.

Diuréticos: Drogas que aumentam a produção de urina.

Diverticulite: Inflamação de divertículos (herniações da mucosa) do intestino grosso.

E.C.G.: Eletrocardiograma . Estudo gráfico da transmissão elétrica do miocárdio (músculo cardíaco).

Edema: Inchaço.

E.E.G.: Eletroencefalograma. Estudo da atividade elétrica do encéfalo.

Extracelular: Espaço localizado fora das células. É formado pelo espaço intravascular (dentro do vaso sangüíneo) e o intersticial (entre as células).

Extrasistoles: Batimentos cardíacos fora de tempo.

Fístula: Comunicação anômala entre duas vísceras ocas ou de entre uma víscera oca e o meio externo.

Glaucoma: Aumento da pressão intraocular.

Hematócrito: Corresponde aos elementos sólidos (glóbulos brancos, glóbulos vermelhos, plaquetas) obtidos por centrifugação do sangue.

Hepatopata: Doente do fígado.

Hipercalemia: potássio sérico acima de 5 mEq/l.

Hipercloremia: Cloro sérico acima de 105 mEq/l

Hiperhidratação: Distúrbio caracterizado por aumento da quantidade de água no organismo.

Hipernatremia: Sódio sérico acima de 145 mEq/l.

Hipocloremia: Cloro sérico abaixo de 95 mEq/l.

Hiponatremia: Sédio sérico abaixo de 138 mEq/l

Homeostasia: Estado de equilíbrio do meio interno.

Iatrogenia: Efeito adverso ao planejado, devido imprudência, descuido ou imperícia do médico quando da instituição do tratamento, acarretando prejuízos para o organismo. O efeito iatrogenico mais grave é representado pela morte do paciente.

Íleo: parada do transito intestina.

Intensivista: Médico que trabalha em UTI (Unidade de tratamento Intensivo).

Interação medicamentosa: efeitos decorrentes do uso de duas ou mais drogas.

LPM: Lista de Procedimentos Médicos. São listados os procedimentos médicos, seus códigos e valores de honorários.

Medicina Nuclear: Ramo da Medicina que utiliza substancias radioativas para realização de procedimentos diagnósticos e terapêuticos.

Miografia: Estudo dos impulsos elétricos dos músculos.

Nefrologia: Especialidade da medicina que estuda o aparelho urinário.

Nefropatia: Doença renal.

Obstrução Intestinal: Situação caracterizada por uma interrupção do transito intestinal que leva a uma parada de eliminação de gases e fezes.

Oligúria: Pouca urina.

Ortostatica: Posição em pé.

Pancreatite: Inflamação do pâncreas.

Polaciúria: Produção excessiva de urina.

Parestesia: dormência.

Sístole: Contração do coração.

Sudorese: Produção de suor.

Suturas: Costura com que se unem os lábios de uma ferida ou incisão.

Taquicardia: Aumento da freqüência cardíaca.

Técnica de Degermação: Conjunto de procedimentos utilizados com a finalidade de eliminação de germes (Limpeza de mãos, uso de anti-sépticos, etc).

Tomografia Computadorizada: Radiografia em série para fixar a um tempo o aspecto de vários planos de um órgão ou região.

Resumo

Apresenta-se neste trabalho a contribuição que os sistemas especialistas poderão oferecer aos estudantes de Medicina, em particular aos de Bases da Técnica Cirúrgica, no sentido de se ter uma ferramenta que permita: simulação de situações clínicas as mais diversas; auxilie no processo de tomada de decisões (diagnóstico e terapêutica); sirva para testar conhecimentos de forma interativa: o computador formula perguntas e o aluno as responde (o aluno verificará se o estado meta (solução de um problema) corresponde ao que ele idealizou).

Como os fluxogramas de decisão correspondem a uma das formas de representação do conhecimento médico mais amplamente utilizadas, desenvolveu-se um software: 'O Sistema Gerador de Regras' (SGR), que permite, a partir do desenho destas estruturas sob forma de grafos valorados, fornecimento de todos os elementos, sobre o formato de um relatório, necessários para construção de sistemas especialistas (variáveis e regras de produção). Para ampliação da contribuição da engenharia do conhecimento na tarefa de elaboração de sistemas especialistas, confeccionou-se, também, uma base de conhecimento que possibilita o diagnóstico de distúrbios hidroeletrolíticos (tópico abordado na disciplina: Bases da Técnica Cirúrgica) que emprega outras formas de representação do conhecimento que não fluxogramas de decisão.

Usando o SGR, os sistemas especialistas poderão ser montados por qualquer usuário, que tenha conhecimento de uso de computadores e interfaces gráficas (Windows), a partir de fluxogramas de decisão previamente elaborados, dispensando na maioria das vezes o engenheiro do conhecimento.

Para a construção dos sistemas especialista empregou-se o "shell" Expert Sinta, desenvolvido pelo Laboratório de Inteligência Artificial (LIA) da Universidade Federal do Ceará.

Abstract

It comes in this work the contribution that the expert systems can offer to the medical students, in matter the one of Surgical Technique Bases, in the sense of having a tool that allows: simulation in most several clinical situations; to aid in the route of to take decisions (diagnosis and therapeutics); to test knowledge in an interactive way: the computer formulates questions and the student answers them (the student will be verified the state goal (solution of a problem) corresponds that he idealized).

As decision's flowcharts correspond one of the medical knowledge representation's ways more thoroughly used, it grew a software: 'Rules Generating System' (SGR), that allows, starting from the drawing of these structures under form of valued graphs, supply all the elements, on the format of a report, necessary to construct expert systems (variables and production rules). To enlarge the contribution of the knowledge's engineering in the task of elaborate expert systems, it was made, also, a knowledge base that makes possible diagnosis in hydro electrolytic disturbances (topic approached in the Bases of the Surgical Technique course) that uses other forms of representation of the knowledge that no decision's flowcharts.

Using SGR, the expert systems can be mounted for any user, that has knowledge in the use of computers and graphical interfaces (Windows), starting from decision's flowcharts previously elaborated, releasing most of the time the knowledge's engineer.

For the expert system construction it was used the Expert Sinta shell, developed by the Laboratory of Artificial intelligence (LIA) of the Ceará Federal University.

Sumário

Lista de figuras

Lista de Tabelas

Introdução

A prática médica é essencialmente baseada na tomada de decisões. São freqüentes os momentos que motivam adoção de atitudes cruciais para o sucesso profissional do médico e para o manuseio dos seus pacientes, tais como:

- Melhor modo de coletar, interpretar, mensurar e registrar dados de anamnese e exame físico do paciente.
- Conclusão do diagnóstico mais provável.
- Avaliação de diagnósticos diferenciais e como excluí-los.
- Decidir pela melhor terapia para o caso.
- Avaliar e detectar complicações da doença e do tratamento evitando interações medicamentosas.
- Determinação do prognóstico: morbidade e mortalidade da doença.
- Definir os custos do tratamento com: exames, medicamentos, materiais, hotelaria, afastamento das atividades laborativas no decorrer da doença e da convalescença, dentre outros.

Rabelo Jr é correto quando observa que o processo de tomada de decisão médica "envolve, portanto, a coleta de dados, o diagnóstico, a recomendação terapêutica e o prognóstico. A coleta de informações consiste na obtenção da história da doença do paciente, dados clínicos e de laboratório. Os dados clínicos compreendem sintomas, que são as sensações descritas pelo paciente, e os sinais, que são manifestações objetivas, alguns mensuráveis, observados pelo clínico. Os resultados de laboratório são descritos como achados. As manifestações referem-se a qualquer sintoma, sinal ou achado. Deste modo conclui-se que o diagnóstico é o processo de manuseio das manifestações clínicas com o objetivo de determinação da doença do paciente" [Rabelo Jr. et al 1993].

Programas de computador podem analisar dados dos pacientes e fornecer informações que auxiliam o trabalho do médico, porém, e isto é importante, sem que atue como substituto do profissional. Na verdade, estes programas de computador que auxiliam no processo de tomada de decisão fazem papel de um médico mais experiente (especialista em Medicina).

Qualquer programa que tenha intenção de auxiliar o médico na tomada de decisão necessita de uma base de conhecimento, que nada mais é do que o conjunto de dados que compõem o conhecimento em uma dada área especifica. Sistemas especialistas são adequados para empenho desta empreitada, à medida que podem armazenar os dados sob o formato de regras de produção e utilizam um motor de inferência para encadeamento destas regras até atingir um objetivo (Um diagnóstico, um tratamento, o exame laboratorial mais adequado, dentre outros).

Uma distinção é feita freqüentemente entre duas formas de aquisição de dados através do computador para programas que objetivem diagnose médica: ativo e passivo.

Na forma ativa, o programa de computador formula perguntas para, a partir daí, obter novas informações do paciente.

No modo passivo é a pessoa usuária do computador que proporciona toda a informação, em um dado ponto, e a diagnose fornecida pelo computador é baseada nesta informação.

O processo ativo tem o inconveniente de que o médico pode estar potencialmente atento para alguns fatos úteis, necessários para o processo de diagnóstico, porém pode não estar habilitado a comunicá-los ao programa, já que, como visto anteriormente, qualquer nova informação deve ser solicitada pelo "software". A abordagem passiva evita este problema, porém a responsabilidade de identificação dos dados pertinentes recai toda ela sobre o médico e como não é esperado que o usuário seja necessariamente um perito no domínio clínico do programa (base de conhecimento) este tipo de restrição é inaceitável.

A representação do conhecimento médico sob o formato de fluxogramas permite que uma sucessão de deliberações seja estruturada no formato de um grafo. Cada nodo representa uma pergunta particular, e a resposta determina qual ramo do grafo deverá ser usado para obtenção da próxima pergunta. Resultados finais são obtidos descendo todo o caminho até chegar a uma folha do grafo.

Embora a árvore de decisão (tipo de grafo) possa ser a estrutura empregada para implementação da imensa maioria dos fluxogramas de deliberação médica, restrições são encontradas na medida em que não serão permitidos: laço em um nó, caminhos duplos, nós com mais de um pai, dentre outras (a Figura 1 exemplifica situação em que um nodo tem mais de um pai).

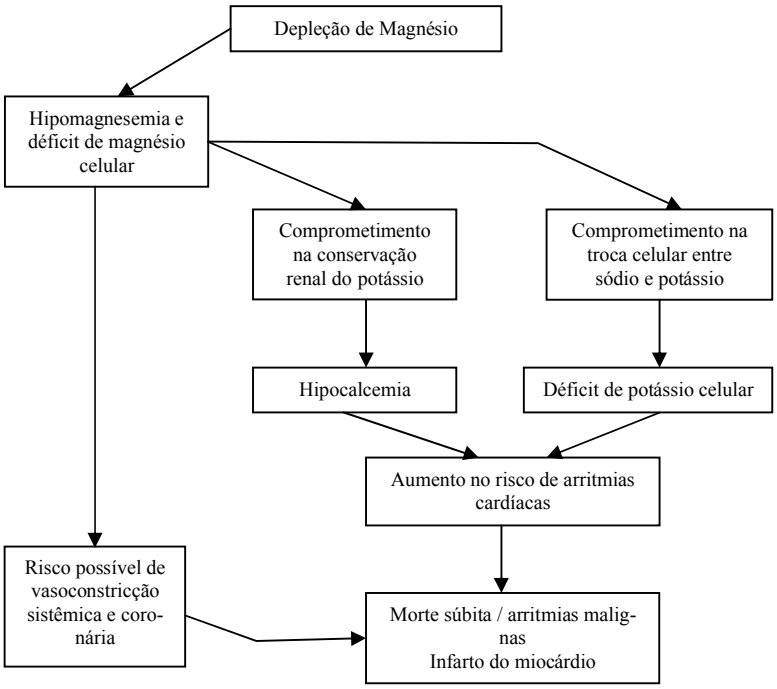


Figura 1 – Relação entre potássio e magnésio [Barruzzi at al 1994].

A estrutura mais adequada que permite implementação de fluxogramas (denominados nos livros médicos algoritmos de decisão) são os grafos valorados. Todos os encaminhamentos no grafo que vão desde o nodo fonte até todas as folhas permitirão formação de regras de produção que possibilitarão a constituição de bases de conhecimento de um sistema especialista.

Os sistemas especialistas podem ser amplamente utilizados no ensino [Nussbaum 2001], contribuindo decisivamente na formação profissional do médico [Villela 1994]. A utilização desses sistemas permite a experimentação: o aluno, sem que nenhum paciente real seja colocado em riscos, poderá cometer erros graves de avaliação e conduta, e dependendo do "software", simular casos não habituais na rotina diária de um hospital ou instituição de ensino.

Hoffer e Barnett assim se expressam "... é evidente que como para praticar a Medicina de forma eficaz, os médicos devem ter acesso rápido ao conteúdo de uma base de conhecimentos médicos grande e complexa, e devem saber como aplicar estes fatos e heurísticas para formular hipótese diagnosticas e para planejar e avaliar terapias, os computadores podem desempenhar um papel direto no processo educacional; os estudantes podem interagir com programas de computador com fins educacionais para adquirir informações

factuais e para aprender a praticar técnicas de solução de problemas médicos" [Hoffer e Barnett 1990].

Objetivos e Motivação desse trabalho

Com o objetivo de facilitar o estudo e resumir determinado tema específico, os autores de livros de Medicina costumam empregar fluxogramas, também denominados algoritmos de decisão, como esquema de representação do conhecimento médico. Os livros de técnica cirúrgica não fogem a regra, e tal construção é usada abundantemente.

Como é feito atualmente, estudar somente a partir do texto contido nos livros torna o processo de aprendizado frio, meramente tarefa decorativa (no sentido de memorização). Então, por que não tornar tal empreitada um procedimento interativo?

A interação poderá ser obtida na medida que, para toda vez que uma decisão precisasse ser tomada, perguntas pertinentes ao problema fossem formuladas pelo aplicativo rodado no computador e o aluno pudesse eleger, para cada questionamento, a resposta que ele considere mais correta. A sucessão de respostas aos questionamentos levará a um estado meta (objetivo do sistema). Se, além disso, para cada pergunta formulada fosse dada a explicação do motivo daquela questão e o raciocínio adotado para obtenção de uma solução (estado meta) fosse fornecido, a lógica adotada pelo aplicativo para resolução de problemas seria mais plenamente justificada e a credibilidade no sistema seria aumentada.

Se a atenção for voltada para o que se pretende obter, constata-se que o objetivo a ser atingido é a construção de programas de computador que interagindo com o usuário permitam que tarefas decisórias sejam solucionadas e o raciocínio adotado para atingir um dado objetivo seja justificado, ou seja, na verdade o que se pretende arquitetar são sistemas especialistas.

Uma das frases mais usada em construção de softwares é procurar não "reinventar a roda", significando que, no processo de elaboração de programas, deve-se evitar partir do zero, e para isto, faz-se o reaproveitamento de algoritmos e rotinas previamente criados.

Não teria sentido, todas as vezes que houvesse intenção de produzir sistemas especialistas, específicos para determinada área do conhecimento, o desenvolvedor tivesse que construir toda a estrutura de tais sistemas. Existem prontos "shells" de sistemas especialistas, alguns "freeware", que permitem que a preocupação única seja a elaboração de bases de conhecimento, atitude que facilita enormemente a empreitada de construção de tais sistemas.

O responsável pela construção de base de conhecimento dos sistemas especialistas é o engenheiro do conhecimento. O que fazer quando este profissional não é disponível?

O objetivo deste trabalho é mostrar que é possível a construção de sistemas especialistas, baseados em fluxogramas de decisão, para que o aluno possa aprender mais facilmente a disciplina na qual o autor é professor e coordenador, na Universidade Federal de Sergipe: Bases da Técnica Cirúrgica.

Para que isto seja possível, foi construído um programa que atua como auxiliar, e em algumas vezes opera como substituto do engenheiro do conhecimento na empreitada de construção de sistemas especialistas, elaborando fatos e heurísticas a partir de fluxogramas de decisão, implementados sob a forma de grafos valorados, que permite a alimentação de qualquer "shell", baseados em regras de produção, destinado à construção de sistemas especialistas.

Estrutura da dissertação

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

No Capítulo 1, apresenta-se um breve histórico da Inteligência Artificial (ou IA), abordando conceitos que permitirão situá-la no contexto das ciências da computação. Apresentam-se ainda, de forma bem mais detalhada, técnicas de representação do conhecimento com ênfase no sistema de representação a partir de regras de produção. O trabalho do engenheiro do conhecimento também é enfatizado neste capítulo.

O Capítulo 2 trata mais especificamente dos sistemas especialistas, apresentando-se aspectos relacionados a sua construção, aplicações, componentes. Alguns sistemas especialistas mais conhecidos na área médica são descritos.

O Capítulo 3 descreve em linhas gerais um programa, o 'Sistema Gerador de Regras' (SGR), elaborado com o objetivo de a partir de fluxogramas de decisões, um dos esquemas mais empregadas para resumir o conhecimento médico e traçar condutas, obter os fatos e heurísticas para alimentar qualquer "shell" de sistema especialista (baseados em regras de produção).

O Capítulo 4 trata do ambiente de desenvolvimento do 'Programa Gerador de Regras', funcionando como manual do usuário.

O Capítulo 5 apresenta a implementação de dois sistemas especialistas, dentre os inúmeros que serão utilizados para possibilitar o aprendizado dos alunos da disciplina 'Ba-

ses da Técnica Cirúrgica'. O primeiro construído integralmente com o 'Sistema Gerador de Regras' (SGR). O segundo, somente parte dele foi elaborado com a ajuda do SGR, já que, foi necessária a ajuda de técnicas de engenharia do conhecimento objetivando elaboração de parte das regras (O raciocínio adotado para construção das mesmas será explicado).

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

A dissertação traz ainda dois anexos: o primeiro corresponde ao material didático adotado pela disciplina '*Bases da Técnica Cirúrgica'* para ilustração das aulas que abordam o tema 'Distúrbios Hidroeletrolíticos', que serviu de apoio para construção do sistema especialista para diagnóstico das alterações hidroeletrolíticas mais freqüentes. O segundo anexo traz o código fonte do *Sistema Gerador de Regras*.

Com o intuito de facilitar a leitura para pessoas não habituadas com a linguagem adotada na pratica médica, foi elaborado um glossário contendo os termos médicos empregados na dissertação.

Capítulo 1
Inteligência Artificial

1.1 Inteligência Artificial - Generalidades

1.1.1 Definição de Inteligência Artificial

Dentre outros conceitos pode-se definir a Inteligência como a parte 'computacional' relacionada à habilidade para alcançarem-se objetivos no mundo real. Inteligência é, também definida, como a habilidade de realizar de forma eficiente uma determinada tarefa. Segundo o dicionário Aurélio a "Inteligência é a faculdade ou capacidade de aprender, compreender ou adaptar-se facilmente; maneira de entender ou interpretar" [1].

Diversos tipos e graus de inteligência ocorrem nas pessoas, nos animais e até mesmo em algumas máquinas. Etimologicamente a palavra inteligência vem do latim inter (entre) e legere (escolher), significando, portanto, aquilo que nos permite escolher entre uma coisa e outra. A palavra artificial vem do latim artificiale, significando algo não natural, isto é, produzido pelo homem. Baseado nestes princípios, Inteligência Artificial (IA) é um tipo de inteligência, produzida pelo homem, para dotar as máquinas de algum tipo de habilidade que simula a inteligência humana.

Diariamente, deparamos-nos com muitos afazeres, que poderiam ser executados razoavelmente valendo-se apenas da nossa inteligência - como, por exemplo, aritmética complexa – tarefas que computadores podem fazer com muita facilidade, se programados para tal. Por outro lado, muitas tarefas que as pessoas fazem intuitivamente - tais como: reconhecimento de uma pessoa, entendimento de imagem visual, reação rápida perante uma nova situação, solução de problemas na ausência de informação completa - são extremamente complexas para automatizar em um programa de computador. A IA está relacionada a estas tarefas difíceis que necessitam de complexos e sofisticados processos de raciocínio e conhecimento.

[1] pág. 774 do Novo Dicionário AURÉLIO.

Criar uma máquina semelhante ao homem, capaz de pensar, sempre foi um fascínio que vem atraindo cientistas de todo mundo obstinados à concretização deste sonho. Desde os tempos mais remotos, escritores já esboçavam robôs e andróides em suas obras de ficção. Atualmente é possível encontrar alguns sistemas e agentes capazes de simplificar, melhorar e aumentar a produtividade em tarefas outrora exclusivas a seres humanos.

As máquinas, denominadas inteligentes, já existem, entretanto estão restritas a laboratórios e centros de pesquisas e, apesar de todos os avanços alcançados nesta área, os humanos estão ainda muito distantes da criação de uma máquina capaz de pensar, como é costumeiro observar nas obras literárias de ficção científica (2001 – Uma Odisséia no Espaço, O Exterminador do Futuro, The Jetson, Guerra nas Estrelas, Inteligência Artificial dentre tantas outras).

É muito difícil, saber exatamente quando se iniciou a área de pesquisa que hoje se denomina Inteligência Artificial, pois, desde os primórdios, o homem sempre sonhou em ser criador e não somente uma criatura. Além disto, os primeiros registros de seres artificiais são místicos, ou mesmo lendários, o que impossibilita de separar nitidamente a imaginação da realidade.

Os primeiros estudos, sobre o que viria a ser chamada Inteligência Artificial, surgiram na década de 40, período marcado notavelmente pela II Guerra Mundial. Esta guerra gerou a necessidade de desenvolvimento de tecnologias voltadas para a análise de balística, quebra de códigos e cálculos para projetar bombas atômicas. Surgia, desta maneira, os primeiros projetos de construção de computadores, assim chamados por serem máquinas utilizadas para realização de cálculos.

O termo “Inteligência Artificial” foi cunhado em 1956 num encontro de cientistas realizado em Dartmouth. Dentre os presentes a este encontro destacavam-se: Allen Newell, Herbert Simon, Marvin Minsky, Oliver Selfridge e John McCarthy.

No final dos anos 50 e inicio dos anos 60, os cientistas Newell, Simon, e J. C. Shaw introduziram o processamento simbólico. Ao invés de construir sistemas baseados em números, eles tentaram construir sistemas que manipulassem símbolos. A abordagem era poderosa e foi fundamental para muitos trabalhos posteriores.

Dentre as várias definições de IA, destaca-se:

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

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

"O estudo das faculdades mentais através do uso de modelos computacionais" [Charniak 1985]

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

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

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

"O ramo da Ciência da Computação que se preocupa com a automação do comportamento inteligente". "Ramo da Ciência da Computação dedicado à automação de comportamento" [Luger 1993]

"Uma área de pesquisa que investiga formas de habilitar o computador a realizar tarefas nas quais, até o momento, o ser humano tem um melhor desempenho"[Rich 1994]

Nenhuma definição de IA, como as que se viu acima, é universalmente aceita, mas pode-se chegar a um conceito eqüidistante para a grande maioria dos escritores: "IA é a área de estudos voltada para a produção de sistemas artificiais que realizam tarefas que, quando realizadas por seres humanos, exigem inteligência". Esta definição é bastante efêmera e esconde a sua abrangência, sua característica interdisciplinar e a enorme complexidade dos problemas presentemente enfrentados; apesar disso, esta definição indica claramente seu objetivo e a peculiaridade de sua natureza.

A Tabela 1. 1 compara a Inteligência natural com a Inteligência artificial empregada nos sistemas baseados em regras:

Habilidade	***Inteligência Natural***	***Inteligência Artificial (Sistemas baseados em regras)***
Adquire grande quantidade de informação externa	*Alto*	*Baixo*
Usa sensores	*Alto*	*Baixo*

É criativo ou tem imaginação	*Alto*	*Baixo*
Aprende por experiência	*Alto*	*Baixo*
Retém dados detalhados	*Baixo*	*Alto*
Faz cálculos complexos	*Baixo*	*Alto*
É adaptável	*Alto*	*Baixo*
Usa uma variedade de fontes de informação	*Alto*	*Baixo*
Transfere informação	*Baixo*	*Alto*

Tabela 1. 1 – Diferença entre Inteligência Natural e Artificial (Sistemas baseados em regras)

1.1.2 Objetivo da IA

O objetivo da IA sempre foi o de implementar programas, que de algum modo, pudessem simular o pensamento, isto é, no processo de resolução de uma tarefa qualquer, fosse empregado técnicas que pudessem ser consideradas inteligentes, de sorte que, desse a impressão que tal obra fosse resolvida por um humano. Aliado a isso, existe uma tendência natural em se desejar adquirir computadores para realizar tarefas que requeiram simulação da inteligência humana.

Os estudiosos, dentro da IA, buscam automatizar a inteligência humana por diferentes razões. Uma razão é simplesmente para que possa ser possível entendê-la melhor e baseado neste fato, por exemplo, ser possível testar e refinar teorias psicológicas e lingüísticas escrevendo programas que tentam simular aspectos do comportamento humano. Outra razão, está na possibilidade de que seja possível a obtenção de programas mais inteligentes. A principio, não existe preocupação com a obrigatoriedade de que os programas simulem o raciocínio humano, porém, o estudo da inteligência permite o desenvolvimento de técnicas úteis para resolução de problemas difíceis.

Na década de 60, na tentativa de simular o pensamento humano, os cientistas utilizavam métodos genéricos para resolução de um grande número de problemas, tática que

constituía os chamados programas de propósitos gerais (GPS - Resolutor Geral de Problemas).

Na década de 70, houve uma mudança de rumo. Desenvolver programas de propósitos gerais mostrou-se tarefa trabalhosa e infrutífera. Surgiram, desta forma, programas de propósitos específicos, que utilizavam preceitos baseados em um domínio restrito do conhecimento, denominados 'Sistemas Especialistas'.

Sistemas especialistas são programas de computador que utilizam técnicas de Inteligência Artificial, tais como: representação simbólica, inferência e busca heurística no sentido de realizar tarefas sofisticadas normalmente possíveis somente para especialistas humanos.

Na década de 80, a Inteligência Artificial torna-se uma verdadeira indústria com destaque para o Projeto de 5ª geração, desenvolvido no Japão, com pesquisas relacionadas à construção de computadores inteligentes utilizando Prolog. Da mesma época merece destaque o desenvolvimento de sistemas de visão robótica.

Presentemente, a IA busca desenvolvimento, apoiado sobre técnicas já existentes, baseadas em teoremas rigorosos e evidências empíricas fortes e não apenas em intuição e, com isso, procura dá maior importância a problemas do mundo real (não apenas "toy problems").

O objetivo final da IA é embarcar em diferentes sistemas/produtos resolvendo problemas de difícil resolução através de técnicas convencionais. Dois grandes enfoques são empregados para tal objetivo:

- Simbólico: dá ênfase aos processos cognitivos, ou seja, a forma como o ser humano raciocina. Objetiva encontrar uma explicação para comportamento inteligente baseado em aspectos psicológicos e processos algorítmico. Realiza processamento de símbolos (Sistemas especialistas, Processamento de linguagem natural, etc).
- Conexionista: Também denominada de biológica ou ascendente, dá ênfase no modelo de funcionamento do cérebro, dos neurônios e das conexões neurais. Representado por aplicações matemáticas (Redes Neurais, etc).

A cada dia que passa, aumenta o número de trabalhos e pesquisas a procura de ferramentas computacionais em Inteligência Artificial que buscam capturar e simular o desempenho de especialistas humanos, tarefa esta que tanto pode ser desenvolvida a partir do zero, com a utilização de linguagem de programação, ou a partir de programas já existentes,

como acontece, por exemplo, com os chamados “shells” geradores de Sistemas Especialistas.

1.1.3 Classificação dos sistemas de IA

Pode-se dizer que a IA tem duas abordagens:

- Simulation mode: tentativa de se reproduzir o comportamento humano frente a um determinado problema.
- Performance mode: o que se busca é desempenho inteligente na resolução do problema (tempo gasto para obtenção do estado meta).

Russell classifica os Sistemas de IA baseados nos termos "pensar" e "agir" sobre os modelos “humanos" e "racional" nas quatro seguintes categorias [Russell 1995]:

- Sistemas que pensam como seres humanos: esses sistemas expressam a abordagem cognitiva e são na sua grande maioria baseados em modelos oriundos dos experimentos psicológicos, estudados pelas ciências cognitivas. Tais modelos são construídos com base nas investigações experimentais em seres humanos (e alguns outros animais), e representam uma das mais importantes contribuições da IA, embora a construção desse tipo de sistemas não seja seu objetivo.
- Sistemas que agem como seres humanos: essa abordagem é bem representada pelo clássico *Teste de Turing* [Turing 1950], elaborado para identificar comportamento inteligente, onde uma pessoa interroga duas entidades, uma humana e uma artificial (máquina), através de uma interface comum. O sucesso viria com a incapacidade do interrogador em decidir qual entidade seria a máquina. A IA não tem se dedicado a tratar desse tipo de abordagem, pois a expressão da inteligência não existe apenas na interação com humanos.
- Sistemas que pensam racionalmente: essa é a abordagem lógica; a abordagem do "pensamento correto". Baseia-se na utilização da lógica formal, que fornece um modelo preciso para as coisas que existem no mundo e as relações entre as mesmas. O desenvolvimento dessa abordagem tem encontrado alguns obstáculos devido a restrições computacionais e à dificuldade de tradução do conhecimento informal para expressá-lo em termos formais. Não se pode, porém, deixar de incluir esta abordagem como uma das mais importantes, devido ao seu poder de representação e de raciocínio.

- Sistemas que agem racionalmente: essa é a abordagem dos agentes racionais; a abordagem das crenças e dos objetivos. Agir racionalmente significa alcançar um objetivo de acordo com uma crença. Um agente é algo que percebe e age. Esta abordagem é indiscutivelmente a mais recente da IA e sua importância é tão grande que alguns autores chegam a redefinir a IA como a área dedicada à construção de agentes racionais.

1.1.4 A Inteligência Artificial é possível?

Automatizar a inteligência humana é realmente uma tarefa factível?

Pesquisas realizadas na área da inteligência artificial fazem supor que a inteligência humana pode ser reduzida à (complexa) manipulação de símbolos e o meio usado para manipulá-la não precisa ser necessariamente um cérebro biológico [Causey 1994].

Discute-se que a verdadeira inteligência nunca poderá ser alcançada por um computador, devido à constatação óbvia de ser imprescindível a utilização de alguma propriedade humana que não pode ser simulada por meios tecnológicos, pelo menos, com os recursos atualmente disponíveis. A partir daí várias posições poderiam ser adotadas [Causey 1994]:

- Computadores nunca serão realmente inteligentes, entretanto eles poderão fazer algumas tarefas úteis que convencionalmente requerem inteligência.
- Computadores poderão eventualmente ser inteligente, porém na realidade o que existe é a simulação de um comportamento inteligente, não sendo, portanto, realmente inteligentes.
- Computadores serão, eventualmente, realmente inteligentes.
- Computadores não só serão inteligentes como serão cônscios e terão emoções.

O que se observa, baseado na atualidade, é que na verdade computadores podem se comportar claramente de forma inteligente quando executam determinadas tarefas. No entanto, ainda na maioria das vezes, de certa forma, de modo limitado. Está claro que poderão ser usadas técnicas de IA para produzir programas úteis que convencionalmente requeiram inteligência humana, e que o processo de elaboração desses "softwares" contribuem para o entendimento da natureza da nossa própria inteligência.

Mesmo com a enorme quantidade de técnicas de IA disponíveis atualmente, a capacidade das máquinas inteligentes (geralmente computadores), independentemente da ação humana, ainda é bastante limitada (A IA mal saiu do seu estágio embrionário e os seus sis-

temas são capazes apenas de manipular inteligência em níveis rudimentares). Mesmo assim, não se pode deixar de antever as infinitas possibilidades existentes para que estas máquinas venham a demonstrar um comportamento inteligente comparável, e, em alguns casos, superior ao humano.

1.1.5 Aplicações da IA

- Visão: A habilidade de dar sentido ao que se enxerga. Busca desenvolver formas para o computador trabalhar com a visão bidimensional e tridimensional.
- Compreensão da linguagem natural: A habilidade para comunicação com os outros, em português ou outro idioma natural, podendo também possibilitar tradução de um idioma para outro. Objetiva aperfeiçoar a comunicação entre as pessoas e os computadores.
- Compreensão da voz: A habilidade para que a interação homem / máquina se processe através da voz.
- Planejamento: A habilidade para decidir, por uma sucessão boa de ações, e alcançar metas pré-estabelecidas.
- Robótica: Permite que um robô possua habilidade para mover e agir no mundo, ao mesmo tempo em que possibilita resposta automática, quando diante de percepções novas. É o campo de estudo voltado para desenvolver meios de construir máquinas que possam interagir com o meio (ver, ouvir e reagir aos estímulos sensoriais). A expressão robô vem do tcheco "robota", significa trabalhador. Foi criada por Karel Capek, na Tchecoslováquia, em 1917. O primeiro robô industrial do mundo, batizado de UNIMATE, surgiu em 1962 [Firebaugh 1988].
- Sistemas especialistas ou sistemas baseados no conhecimento: Habilidade de simular conhecimento só possível a um especialista em determinada área do conhecimento. Devido ao fato do computador ser especialmente útil para automatizar tarefas e como há uma escassez de peritos humanos (médicos, engenheiros, economistas, dentre outros) em determinadas áreas geográficas do planeta, Sistemas Especialistas inclui dentre outras habilidades:
    - Diagnose médica.
    - Conserto de equipamento.
    - Configuração de computador.
    - Planejamento financeiro.

- Manipulação de jogos de estratégia (Xadrez, Poker, Damas, etc): É o estudo voltado para a construção de programas de jogos envolvendo raciocínio. Os jogos computadorizados são um grande sucesso, ainda mais quando exibem um tipo de inteligência capaz de desafiar as habilidades do jogador.

1.1.6 Disciplinas da IA

As disciplinas de Inteligência Artificial são matérias de estudo bem definidas, cada uma delas abordando um assunto específico que está relacionado com a área de IA. Apresenta-se na fig. 1. 1 os principais métodos ou técnicas de IA atualmente abordados [Carvalho 99].

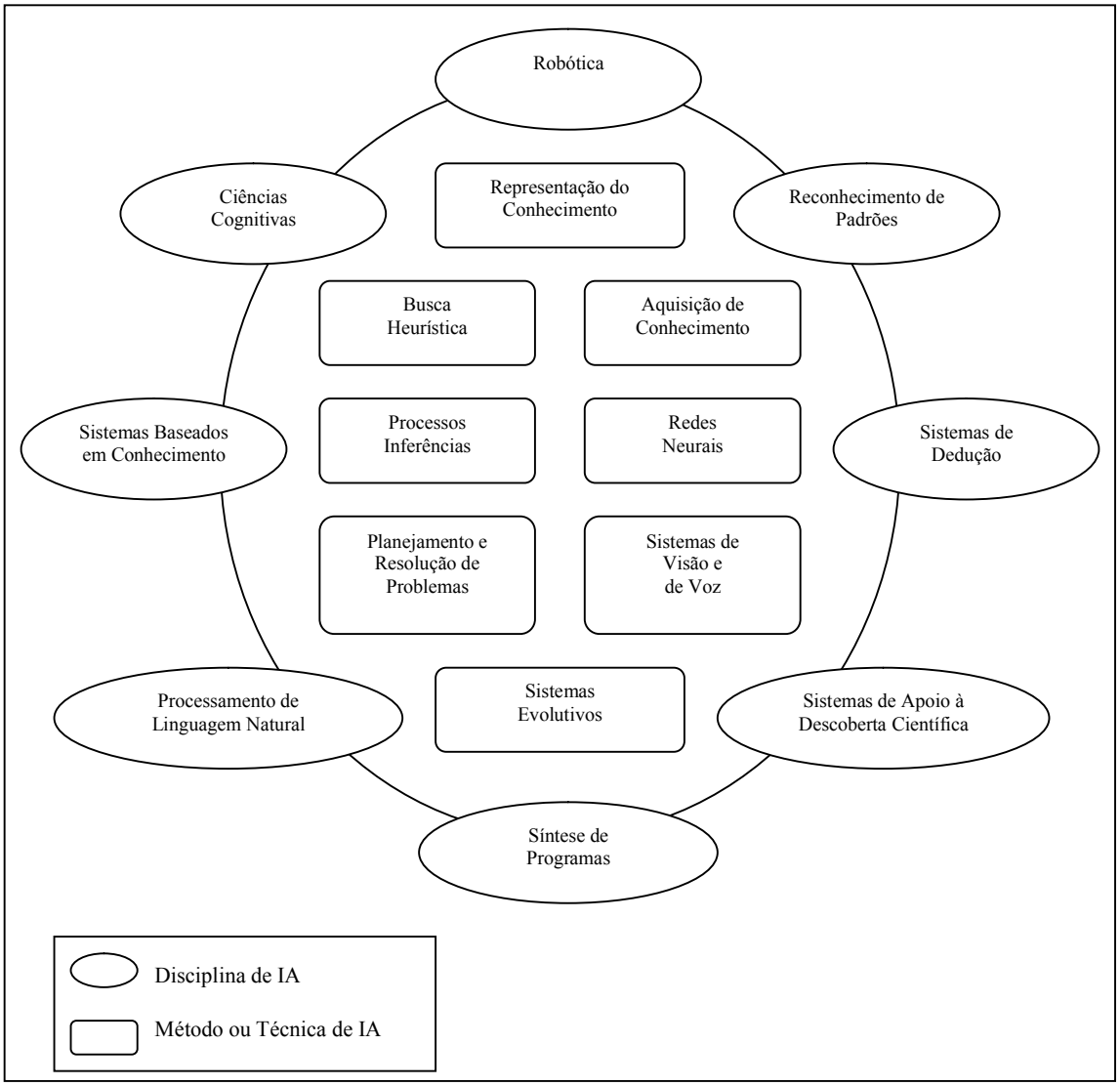


Figura 1. 1 - Métodos e Disciplinas da Inteligência Artificial

1.2 Técnicas empregadas em IA

Diversas técnicas podem ser aplicadas a uma variedade de tarefas de IA. Estas técnicas estão preocupadas com a maneira como se representa, manipula-se e argumenta-se o conhecimento para resolução de problemas.

1.2.1 Representação do conhecimento

Problemas relacionados com a representação do conhecimento não surgiram com o advento dos computadores digitais. Há séculos têm sido estudados por filósofos [Alfredo 2001], cujos trabalhos jamais puderam ser ignorados pelos cientistas da computação. Para os pesquisadores em IA, estes problemas continuam sendo bastante significativos.

A representação do conhecimento é tarefa crucial. Um dos resultados mais claros da pesquisa da inteligência artificial é que, para resolução de problemas aparentemente simples, é requerido muito conhecimento. Realmente, entender uma única frase requer um conhecimento extenso do idioma e do seu contexto. Problemas que envolvem um determinado domínio particular geralmente requerem conhecimento dos objetos do domínio e conhecimento de como argumentar dentro daquele domínio – deve-se procurar maneiras de representar ambos os tipos de conhecimento.

O conhecimento deve ser representado eficazmente e de um modo significativo. Eficiência é importante, por isso é impossível (ou pelo menos não prático) representar todo fato explicitamente. Tem-se que ser capaz de deduzir fatos novos a partir do conhecimento existente.

O conhecimento deve ser adequadamente representado de forma que se possa relacioná-lo ao mundo real. Um esquema de representação de conhecimento deve prover um mapeamento de características do mundo para um idioma formal. Isto é o que se quer dizer quando se aborda a semântica de idiomas de representação.

Na representação do conhecimento deve-se, portanto, usar técnicas que:

- Capturem generalizações acerca do que se quer representar.
- Possam ser compreendidos por todas as pessoas que manuseiem o conhecimento (aquisição e processamento).
- Possa ser facilmente modificado.
- Possa ser utilizado em diversas situações, mesmo que não seja totalmente exato ou completo.
- Possa ser utilizado para reduzir a faixa de possibilidades, que, usualmente deveria se considerar para buscar soluções.

O conhecimento declarativo pode ser representado com modelos relacionais e esquemas baseados em lógica. Os modelos relacionais podem representar o conhecimento em forma de árvores, grafos ou redes semânticas. Os esquemas de representação lógica incluem o uso de lógica proposicional e lógica dos predicados.

1.2.2 Busca

Outra prática geral crucial requerida para elaborar programas de IA é à procura de uma solução. Freqüentemente não há um caminho direto para achar uma solução para um determinado problema. Porém, é possível gerar caminhos para atingir metas pré-estabelecidas. Por exemplo, na resolução de um quebra-cabeça pode-se saber todos os possíveis movimentos, mas não a sucessão que vai conduzir a um estado meta.

Um problema em IA é definido em termos de um espaço de estados possíveis, incluindo: um estado inicial e um (ou mais) estado final que é o objetivo a ser atingido e um conjunto de ações (ou operadores) que permitem passar de um estado a outro.

As técnicas que fazem uso de força bruta, ou seja, aquelas para as quais são geradas e experimentadas todas as possíveis soluções para alcançar um objetivo, são freqüentemente muito ineficientes, pois são muitos os espaços de busca possíveis.

Os processos de busca que explora todas as condições podem ser:

Em profundidade: Utiliza menos memória. A busca é realizada por acaso (depende da ordenação dos sucessores) e pode encontra a solução sem examinar grande parte do espaço de busca.

O algoritmo da busca em profundidade pode ser esquematizado como [Rich at al 1994]:

1. Se o estado inicial é um estado de meta, saia e retorne sucesso.
2. Caso contrário, faça o seguinte até a sinalização de sucesso ou fracasso:
    a. Gere um sucessor, E, do estado inicial. Se não houver mais sucessores, sinalize fracasso.
    b. Chame busca em profundidade com E como estado inicial.
    c. Se for retornado sucesso, sinalize sucesso. Caso contrário, continue nesse laço (“loop”).

A figura 1. 2 mostra esquematicamente a árvore de busca em profundidade.

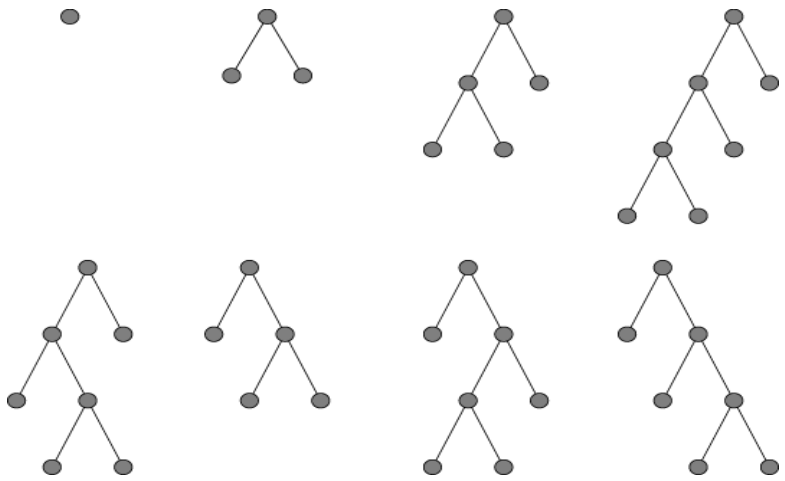


Figura 1. 2– Busca em profundidade

Esta estratégia deve ser evitada quando as árvores geradas são muito profundas ou geram caminhos infinitos. Para problemas com várias soluções, esta estratégia pode ser bem mais rápida do que a busca em largura.

Em largura: não pesquisará becos sem saída. Se houver uma solução, ela será encontrada. Se houver várias soluções, então uma solução mínima será encontrada (tamanho do caminho).

O algoritmo utilizado na busca em largura é [Rich at al 1994]:

1. Crie uma variável chamada Lista-de-Nós e ajuste-a para o estado inicial.
2. Até ser encontrado um estado-meta ou Lista-de-Nós ficar vazia, faça o seguinte:
    a. Remova o primeiro elemento de Lista-de-Nós e chame-o de E. Se Lista-de-Nós estiver vazia, saia
    b. Para cada maneira como cada regra pode ser casada com o estado descrito em E, faça o seguinte:
        I. Aplique a regra para gerar um novo estado.
        II. Se o novo estado for um estado-meta, saia e retorne este estado.
        III. Caso contrário, acrescente o novo estado ao final de Lista-de-Nós.

A figura 1. 3 mostra os nós que são gerados sucessivamente a partir do nó fonte.

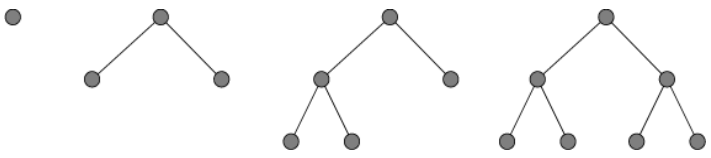


Figura 1. 3– Busca em largura

A Figura 1. 4 mostra de maneira mais clara os passos utilizados para a realização da busca em largura. O nodo preenchido (preto) é o estado meta e a busca pára quando da geração deste nodo.

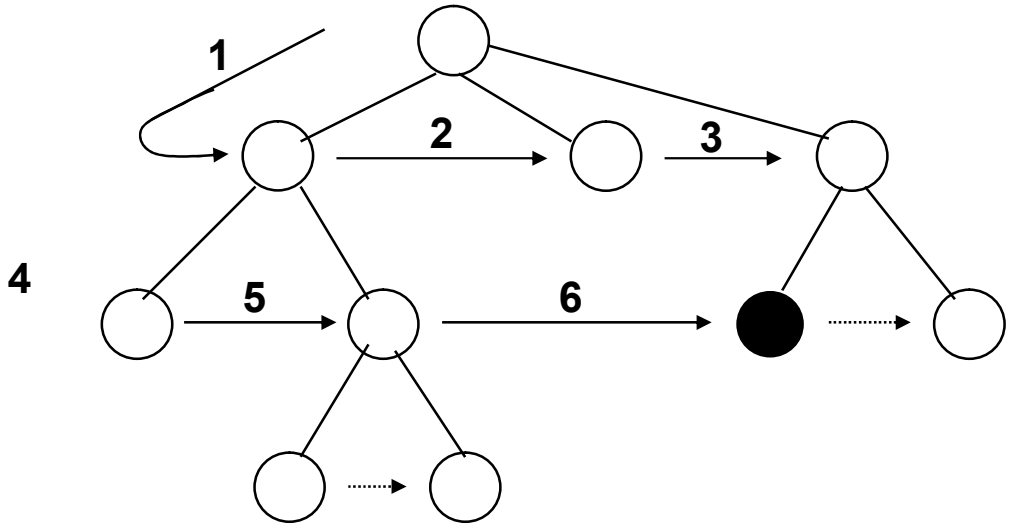


Figura 1. 4– Busca em largura

Para os processos que objetivam resolução de problemas em IA, surgem, como se começa a perceber, possibilidades de decisão que podem ser representadas por árvores. Os métodos de busca e o uso de heurísticas indicam qual o ramo a ser seguido de forma a tomar o menor caminho possível na procura da solução.

Técnicas heurísticas são freqüentemente melhores, pois só são tentadas as opções que se acha (baseado na melhor suposição atual) ser provável conduzir a uma solução boa, não necessariamente a melhor, implicando com isto, uma economia considerável de tempo no processo de busca.

Uma das dificuldades das heurísticas é que, algumas vezes, duas heurísticas fazem recomendações contraditórias. Sabendo-se que uma heurística é melhor que outra se deve dar prioridade a melhor. Mas o que fazer quando as duas parecem boas? Neste caso é melhor dar um valor numérico para os estados sucessores de um dado estado e decidir continuar a busca pelo sucessor que tem o "melhor" valor. O método para calcular estes números é denominado função de avaliação (FA).

Por convenção, os valores assumidos por uma função de avaliação são números não negativos tal que o estado associado com o menor número é considerado o estado mais promissor. Freqüentemente, a função de avaliação do estado meta é zero.

Além das funções de avaliação há outras funções que podem auxiliar o processo de busca, denominadas funções de custo (FC). Elas são funções não negativas que medem a dificuldade de ir de um estado para um outro. Usando estas funções é possível encontrar não somente um caminho, mas um bom caminho ou ainda o melhor caminho para alcançar uma dada meta.

A fim de não confundir funções de custo com funções de avaliação, é importante observar que as funções de custo se referem ao passado, enquanto que as funções de avaliação se referem ao futuro. Isto é, as funções de avaliação "adivinham" quão perto está um estado do estado meta, enquanto que as funções de custo "sabem" quão longe está um estado do estado inicial. Nesse contexto, as funções de custo são mais concretas que as funções de avaliação.

Varias estratégias de busca heurísticas poderão ser utilizadas conforme evidencia a Tabela 1.2 (O termo agenda refere-se à organização das trajetórias em uma fila de espera):

Estratégia de busca	**Usa agenda?**	**Usa FA?**	**Usa FC?**	**Próximo estado**
Profundidade	*Não*	*Não*	*Não*	*O sucessor do último estado, caso contrário, o sucessor do predecessor.*
Largura	*Sim*	*Não*	*Não*	*O estado mais longe na agenda (fila)*
Hill-Climbing	*Não*	*Sim*	*Não*	*O sucessor com mínimo valor da FA*
Best-First	*Sim*	*Sim*	*Não*	*O estado da agenda com mínimo valor da FA*
Branch-and-Bound	*Sim*	*Não*	*Sim*	*O estado da agenda com mínimo valor da FC*
*A**	*Sim*	*Sim*	*Sim*	*O estado da agenda com mínimo valor da soma de FA e FC*

Tabela 1. 2 – Métodos de Busca heurística – Comparação

A Solução de problemas, usando técnicas de busca heurística, pode apresentar dificuldades em definir e usar a função de avaliação (FA), principalmente quando não consideram conhecimento genérico do mundo (ou "senso comum"). A função de avaliação apresenta compromisso na definição do tempo gasto na seleção de um nó e a redução do espaço de busca, o que faz com que achar o melhor nó a ser expandido a cada passo pode ser tão difícil quanto o problema da busca em geral.

1.2.3 Programação em Inteligência Artificial

Programas em IA, em princípio, podem ser desenvolvidos em qualquer linguagem de programação, porém, como em qualquer tarefa de programação, há linguagens que apresentam características mais adequadas a esta finalidade. As linguagens mais adequadas são aquelas que fornecem apoio à computação simbólica e que sejam inerentemente exploratórias.

Apoio para Computação Simbólica:

Programação em IA envolve principalmente a manipulação de símbolos, e não, como ocorre na programação convencional, de números. Estes símbolos podem representar objetos do mundo e relações entre esses objetos.

Na representação simbólica freqüentemente empregam-se listas, como estrutura de dados para representar o conhecimento, na qual um elemento dessa lista pode ser ou um símbolo ou outra lista. Linguagem para programação em IA, portanto, devem apoiar estruturas de dados baseadas em listas.

Apoio para Programação Exploratória

Para muitos, senão a maioria, programas de IA exibem problemas relacionados a técnicas de engenharia de software. Raramente é possível dar a especificação completa de um programa de IA antes da construção de um protótipo que entenda bem a natureza dos problemas. Programar IA é, portanto, inerentemente exploratório.

Características de linguagem de programação que apóia programação exploratória incluem:

- Extensibilidade: habilidade para desenvolver propósitos especiais para resolver classes de problemas;
- Interatividade: a partir do ambientes de desenvolvimento o programador deve ter flexibilidade para testar interativamente seções pequenas do programa.

Principais Linguagens usadas em IA

Existem várias linguagens para a construção de programas de IA. A primeira linguagem para IA surgiu em 1955, com estudos realizados por Newell, Shaw e Simon, denominada de IPL-II. Em seguida surgiu a linguagem LISP (LISts Processing), estruturada

por John McCarthy (em 1958), a qual passou a ser amplamente utilizada juntamente com a lógica matemática e as funções recursivas.

As principais linguagens de programação usadas em IA são o LISP e o Prolog. Ambas têm características que as fazem satisfatórias para programar IA, como: apoio ao processamento de lista, busca de padrões e programação exploratória. Ambas são utilizadas amplamente - Prolog especialmente na Europa e Japão e LISP nos EUA.

LISP pode ser visto como o avô de programação funcional. Desenvolvida no começo da década de 50 tem como característica ser baseado em definições de funções.

LISP usa a lista como fundamento para a representação de estruturas de dados e provê uma gama extensiva de funções para manipulação de listas. (LISP significa: processamento de lista).

Listas permitem que estruturas complexas de símbolos (representando conhecimento em IA) possam ser manipuladas facilmente.

O Prolog é um idioma baseado em lógica. (Prolog = Programando em Lógica). Os primeiros desenvolvedores dessa linguagem incluem Robert Kowalski em Edinburgh (visão teórica), Maarten van Emden em Edimburgo (demonstração experimental) e Alan Colmerauer em Marseilles, França (Implementação). A atual popularidade do PROLOG é grandemente devida a David Warren, pela eficiente implementação em Edimburgo nos meados dos anos 70 [Causey 1994].

O PROLOG é uma linguagem centrada num pequeno conjunto de mecanismos básicos, incluindo casamento de padrões, estrutura de dados baseada em árvores e "backtracking" (retrocesso) automático. Esse conjunto de mecanismos, embora pequeno, constitui uma forma poderosa e flexível de programação.

O PROLOG é especialmente conveniente para tratar problemas que envolvam objetos, especialmente objetos estruturados, e as relações entre esses objetos. Em particular, está baseado em cálculo de predicado de primeira ordem. Escrever programas simples em Prolog é semelhante a escrever declarações em lógica dos predicados para resolução de teoremas [Causey 1994].

Do ponto de vista comparativo são as seguintes às diferenças entre a programação convencional e a simbólica descrita por [Harmon 85]:

Programação Convencional	**Programação Simbólica**
Algoritmos	Heurísticas
Base de Dados acessada numericamente	Base de conhecimento simbolicamente estrutu-

	rada numa memória de trabalho global
Processamento numérico	Processamento simbólico
Processamento seqüencial	Processamento altamente interativo

Tabela 1. 3– Comparação da Programação Convencional com a programação simbólica.

Que esquematicamente pode ser representado como na Figura 1. 5.

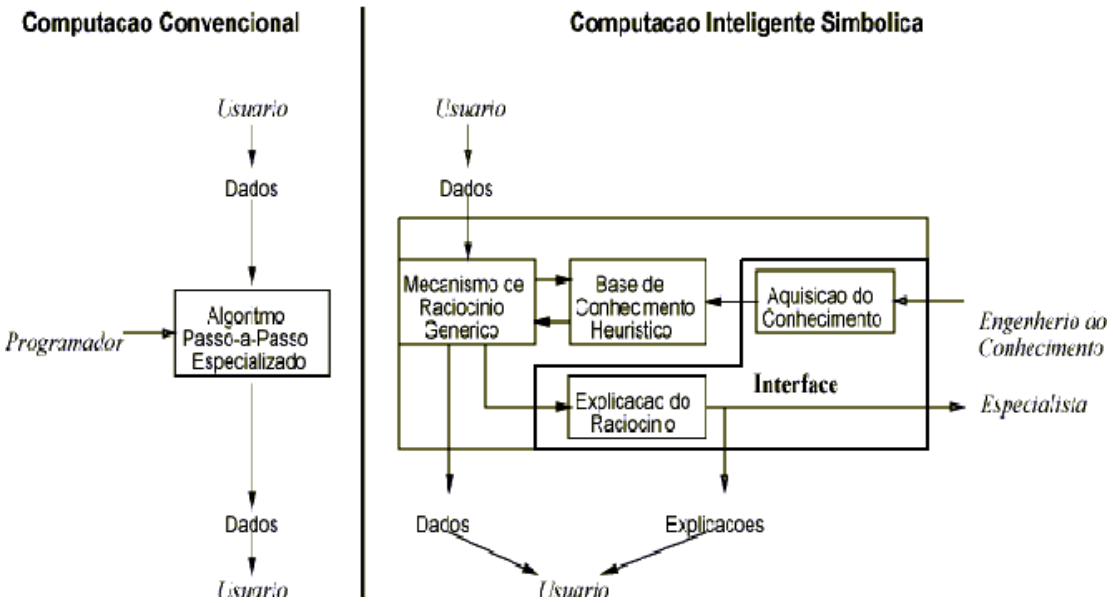


Figura 1. 5 – Programação convencional x Programação Simbólica.

1.3 Representação do Conhecimento

Em Inteligência Artificial comportamentos inteligentes podem ser alcançados pela manipulação de estruturas de símbolos (representando partes do conhecimento). O papel da representação de conhecimento em Inteligência Artificial é o de reduzir problemas de ação inteligente a problemas de busca.

Em IA é crucial que a linguagem de representação do conhecimento deva levar a conclusões. Não se pode representar todo o conhecimento explicitamente, algumas coisas deverão ser colocadas de forma implícita, de modo que possam ser deduzidas pelo sistema quando esse precisa resolver algum problema. Representar tudo explicitamente seria extremamente custoso no que se refere ao consumo de memória.

Em geral, uma boa linguagem de representação de conhecimento deveria ao menos ter as seguintes características:

- Permitir expressar o conhecimento que se deseja representar na linguagem.
- Deveria permitir deduzir conhecimento a partir de um conjunto básico de fatos.
- Deveria tem sintaxe e semântica bem definidas.

Caso contrário não se estaria seguro de que as conclusões seriam corretas ou de que representam à média dos resultados.

Algumas destas características podem estar presentes em linguagem de representação mais recentes não especificas para programação em IA, embora tenham sido influenciadas pela mesma em sua construção, que apresentam como característica serem dedutivas e se valerem de bancos de dados orientados a objeto.

Portanto, para que se tenha uma boa representação do conhecimento necessita-se que a mesma seja:

- Transparente: permitindo o entendimento do que está sendo dito;
- Rápida: possibilitando o armazenamento e a recuperação de informações em tempo curto;
- Computável: possibilitando a sua criação utilizando um procedimento computacional existente.

Na representação do conhecimento, assume importância fundamental a lógica. A lógica, por definição, apresenta sintaxe e semântica bem definidas, e existe preocupação com a verdade para que seja possível obter conclusões. Representar algumas coisas do senso comum em lógica, porém, pode ser tarefa muito complicada. Para estas situações utilizam-se lógicas mais complexas: lógicas temporais e lógicas modais.

A inferência assume primordial importância na lógica, podendo ser por:

- Dedução: a partir de fatos de conhecimento representados de forma lógica, utilizam-se regras de inferência válidas para gerar novos fatos.

  Ex.: Se há fogo, há fumaça.
- Abdução: inverso da dedução.

  Ex.: Se há fumaça, há fogo.
- Indução: parte-se dos fatos até as regras gerais.

Ex.: Se a Carla do Tchan não canta bem, e a Sheila do Tchan também não canta bem, então nenhuma dançarina canta bem.[2]

- Analógico: resolução de problemas baseada em problemas já resolvidos.

  Ex.: Demonstração automática de teoremas

Duas outras formas de representação do conhecimento utilizadas são aquelas que se valem de objetos estruturados e dos sistemas de produção.

A finalidade dos objetos estruturados é representar o conhecimento como uma coleção de objetos e relações. As relações mais importantes são: subdivisão de classe e relações de exemplo. A relação de subdivisão de classe é aquela na qual uma classe é uma subdivisão de outra classe, enquanto a relação de exemplo diz que algum elemento pertence a alguma classe.

Sistemas de produção consistem em um conjunto de regras *se-então*, e uma memória de funcionamento.

1.3.1 Objetos Estruturados

1.3.1.1 Redes semânticas

O tipo mais simples de objeto estruturado é originalmente a rede semântica. Desenvolvida no começo dos anos 60, com a finalidade de representar o significado de palavras inglesas, elas são historicamente importantes devido ao fato de ter introduzido a idéia básica de hierarquias de classe e herança. Originalmente, a idéia de redes semânticas foi proposta em 1913 por Selz como uma explicação de fenômenos psicológicos. Em 1966, Quillian implementou aquelas idéias e mostrou como o significado poderia ser representado como relacionamento entre dois objetos [Causey 1994].

As Redes Semânticas são uma tentativa de se formalizar como nosso conhecimento é organizado na memória.

Uma rede semântica, na realidade, é um grafo onde os seus nodos representam conceitos (objetos ou propriedades), e os arcos representam relações binárias entre conceitos. As relações mais importantes entre conceitos são relações de subdivisão de classe entre classes e subdivisões de classe e relações de exemplo entre objetos em particular e a classe

[2] Grupo baiano do gênero musical "Axé Music" que no final do século 20 tinha como bailarinas: Sheila e Carla.

pai. Porém, qualquer outra relação é permitida, como parte-de, cor etc. As relações mais usadas são:

- "is-a" (é-um): Permite agrupar objetos na mesma classe (Instanciação)
- "ako" (a-kind-of: tipo-de): Refinamento de um conceito em um mais específico (sub-tipagem)
- "part-of" (parte-de): (relação de: pertence a ...)

Pode ser usada a subdivisão de classe e relações de exemplo para derivar informação nova que não é explicitamente representada. Redes semânticas regularmente permitem representação eficiente de conclusões herança-baseadas que usam algoritmos de propósito especiais.

A busca em redes semânticas pode ser usada de várias maneiras para se extrair informações. Por exemplo, a busca pode ser usada:

- Como uma ferramenta explicativa;
- Para explorar exaustivamente um tópico; e
- Para encontrar o relacionamento entre dois objetos.

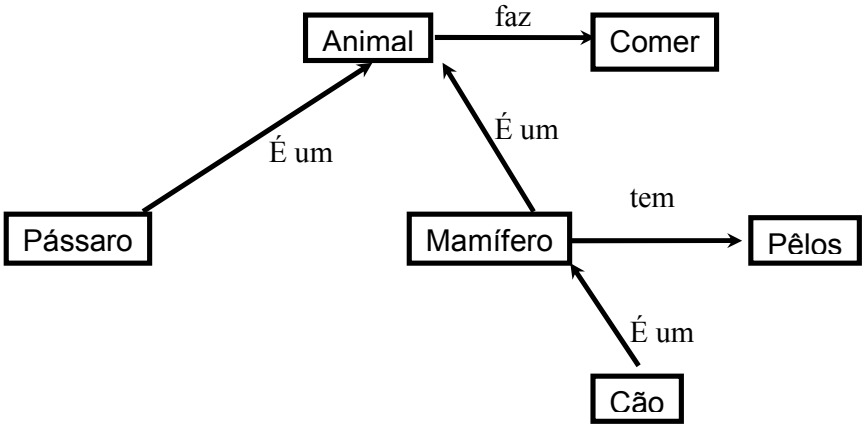


Figura 1. 6– Rede Semântica

No exemplo da Figura 1. 6, para derivar todo o conhecimento sobre cães usa-se busca em largura e, a partir do nó cão encontra-se:

- Cães são mamíferos
- Cães têm pêlos
- Cães são animais
- Cães comem

Para verificar se cães e pássaros estão relacionados executa-se uma busca em largura a partir de ambos os nós e a interseção entre os nós visitados fornece o relacionamento: são animais e comem. Isto é chamado: ativação distribuída ou interseção de busca.

1.3.1.2 Script

Desenvolvido por Schank e Abelson, em 1977, o "script", também denominado roteiro, é uma estrutura que representa o conhecimento descrevendo a seqüência de eventos e fatos presente numa determinada ocasião, de uma forma narrativa. Um "script" usa uma série de campos contendo conhecimento declarativo ou procedural. Condições, objetos, papéis e cenários são componentes típicos de um "script" [Causey 1994].

Em "scripts", os nós são eventos, e os "links" entre eles são simplesmente causais, isto é, um evento provoca o próximo.

"Scripts" são muito similares a "frames", são codificados da mesma forma e são, normalmente, considerados como uma sub classe de "frames".

Exemplo de "script":

Nome: Restaurante

Objetos: mesas, menu, comida, conta, dinheiro.

Agentes: cliente, garçom, cozinheiro, caixa, dono.

Condições de entrada: o cliente tem fome, o cliente tem dinheiro.

Resultados: o cliente tem menos dinheiro, o dono tem mais dinheiro, cliente não tem fome.

Seqüência de cenas:...

Cena 1: Entrada

- Cliente entra no restaurante
- Procura uma mesa
- Decide onde sentar
- Vai para a mesa
- Senta

Cena 2: Pedido

- O cliente pega o menu
- Olha o menu
- Decide o que comer

- Chama o garçom
- Vem até a mesa
- Pede a comida
- ...

O "Script" é empregado para auxiliar no entendimento da estrutura e ordem de eventos.

A partir do argumento: Se Maria foi ao restaurante, comeu uma torta e saiu.
Pode-se inferir sobre Maria que ela:

- Estava com fome
- Tinha algum dinheiro
- Pediu a torta antes de comer
- Um garçom a atendeu
- Pagou antes de sair

1.3.1.3 Frames

"Frames" são uma variante de redes, sendo um dos modos mais populares de representar conhecimento em um Sistema Especialista. Em um frame toda a informação pertinente a um conceito particular é armazenada dentro de uma única entidade complexa. Superficialmente, "frames" se parecem com estrutura de dados de registro, porém, "frames" suportam o conceito de herança.

Um "frame" é organizado de maneira muito semelhante a uma rede semântica. É elaborado como uma rede de nodos e relações organizados numa hierarquia, onde os nodos do topo representam conceitos gerais e os nodos localizados mais abaixo representam instâncias mais especificas destes conceitos. Embora pareça uma rede, num "frame" o conceito de nodo é definido por uma coleção de atributos denominados "slots" ou facetas, e seus respectivos valores. Cada "slot" tem um número qualquer de procedimentos anexados a si, que são executados automaticamente quando a informação contida no "slot" é recuperada ou alterada. "Slots" podem armazenar valores, lista de valores, restrição sobre valores válidos, tipo de dado, indicação de valor não especificado, unidades de medidas, ponteiros.

Procedimentos que estão dentro de "frames" são chamados "demons". Para um "frame" que contenha a representação de um quadrado um exemplo de um "demon" poderia ser dado por um procedimento para calcular a área de um quadrado dado o tamanho de

um dos lados. Deste modo, o valor da área não precisa estar representado podendo ser calculado a partir de outras informações na instanciação do "frame".

A maioria dos sistemas que usam frames como forma de representação do conhecimento distinguem entre valores de atributo típicos e valores definidos que devem ser verdadeiros.

Uma forma surpreendentemente poderosa de conhecimento é representada pelo conhecimento de falta. Este conhecimento é assumido verdadeiro em um "frame" a menos que especifique exceções. Por exemplo, frame de pessoa em um sistema especialista médico pode incluir conhecimento de falta como o seguinte: Para cada pessoa é assumido ter dois braços, dois olhos, dois rins, dois pulmões, e assim por diante. Só para pessoas em que faltem dois de um órgão específico ou parte do corpo existe necessidade de especificação de informação adicional qualquer.

O uso de "frames" pode facilitar a elicitação do conhecimento, por utilizar uma forma de representação de conhecimento similar a utilizado por muitos especialistas, para representação do conhecimento em domínios estruturados como biologia.

Um das formas mais poderosas de argumentar em "frames" é a herança. "Frames" especializados podem herdar propriedades de "frames" mais gerais. Isto permite grandes economias na representação do conhecimento já que muitos "frames" especializados podem herdar as propriedades de alguns "frames" mais gerais sem exigir que essas propriedades sejam reespecificadas em cada exemplo. Objetos na programação orientada a objetos são muito similares aos "frames", devido a este fato. Linguagens orientadas a objetos são boas opções para a implementação de sistemas de frames.

A figura 1. 7 é uma tabela contendo o desenho abstrato de um "frame" e os campos que deverão ser preenchidos neste tipo de representação:

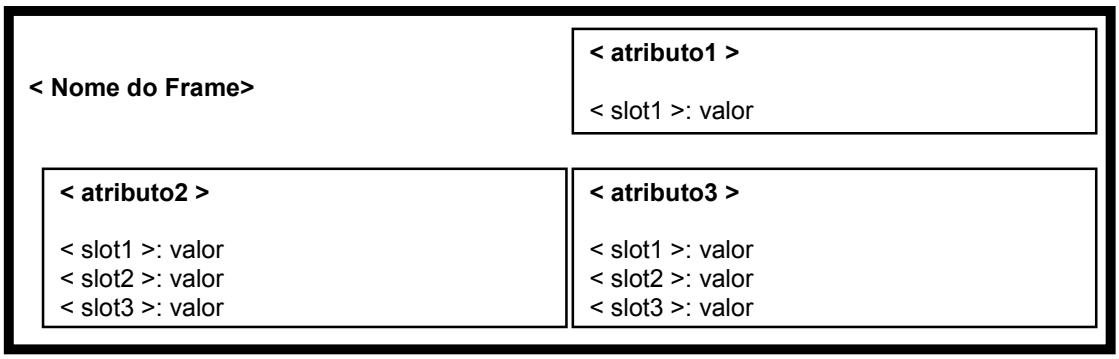


Figura 1. 7– Frame

Avançando na representação a figura 1. 8 mostra um exemplo concreto (Os campos estão preenchidos):

Cômodo	É um: Lugar-coberto		
Atributo	Default	Tipo	Se-necessário
Nº de paredes	4	número	
Formato	retangular	símbolo	
Altura	3	número	
Área		número	
Volume		número	**Área * Altura**

É um

Sala	É um: Cômodo	
Atributo	Default	Tipo
Mobiliário	(sofá,mesa,cadeiras)	lista de símbolos
Finalidade	convivência	símbolo
Área	25	número

Figura 1. 8- Frame

E finalmente um exemplo utilizando uma rede de frames completa (Figura 1. 9):

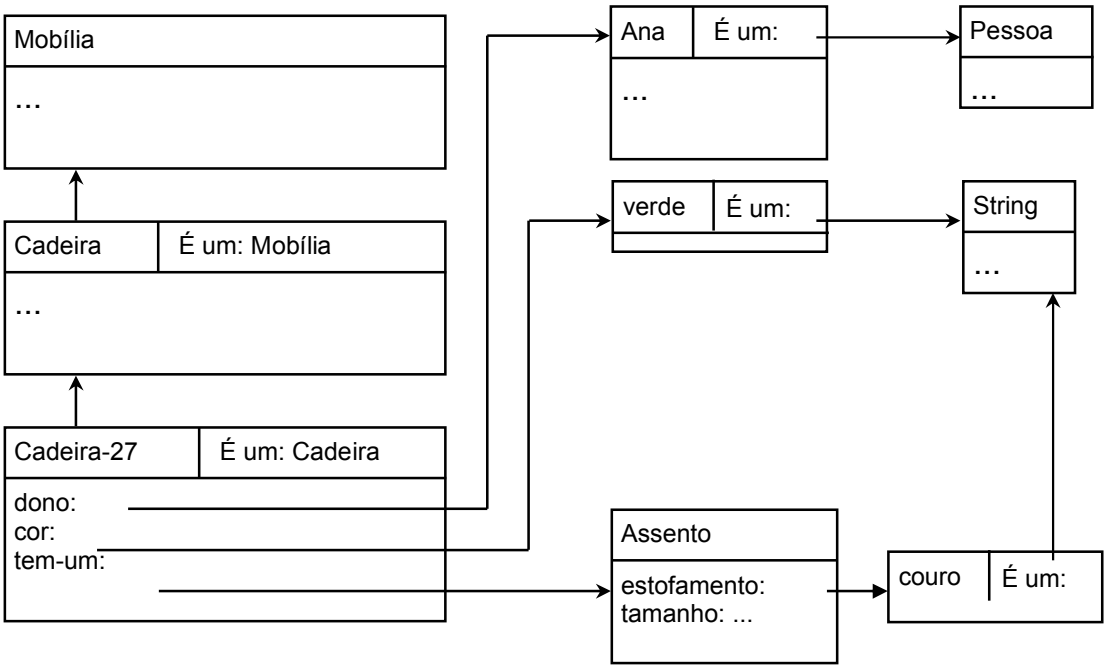


Figura 1. 9– Rede de Frames

1.3.1.4 Lógica dos predicados

A linguagem de representação do conhecimento mais importante é a lógica dos predicados (ou estritamente, lógica dos predicados de primeira ordem). A Lógica dos predicados permite representar fatos complexos acerca do mundo, e, deriva fatos novos de certo modo com garantias que, se os fatos iniciais forem verdadeiros, assim também serão as conclusões. É uma linguagem formal bem entendida, com sintaxe bem definida, semântica e regras de conclusão.

O mundo da lógica dos predicados consiste de objetos, que possuem identidade individual e propriedades que as distinguem de outros objetos. Existem relações entre esses objetos, algumas das quais são funções, ou seja, relações nas quais existe apenas um valor para uma determinada entrada.

Lógica dos predicado pode expressar regras como:

Se < objeto><atributo><valor > então < objeto><atributo><valor > e é capaz de permitir raciocínio sobre os diferentes componentes dessas proposições.

O uso de conectivos lógicos permite construção de sentenças mais complexas.

Símbolos constantes, variáveis e símbolos funcionais são usados para construir termos. Quantificadores e símbolos de predicados são usados para construir sentenças.

Os quantificadores empregados são: o quantificador universal (∀) e quantificador existencial (∃). Símbolo de igualdade (=) é usado para declarar que dois termos referenciam o mesmo objeto.

As formas principais de argumentação em lógica dos predicados são unificação e resolução.

No processo de resolução para provar um teorema/tese, tomam-se duas cláusulas quaisquer, que resolvidas originam uma nova cláusula. Para tanto, essas duas cláusulas devem ter o mesmo literal complementar. Esses dois se cancelam e a cláusula resultante é a que sobrou da anulação. Isso é repetido até que se chegue ao teorema a ser provado.

A lógica dos predicado pode avaliar uma expressão para instanciação das diferentes variáveis e pode reconhecer que pode ser verdadeira para alguns valores e falso para outros. Porque a lógica dos predicado pode usar regras gerais para representar uma gama extensiva de relações específicas, é menos suscetível a uma explosão combinatória na representação do conhecimento que a lógica proposicional.

1.3.1.5 Sistemas baseados em regra

Ao invés de representar o conhecimento de um modo relativamente declarativo, estático (como um grupo de coisas que são verdadeiras), sistemas baseados em regras representam o conhecimento em termos de um grupo de regras que apontam para o que deve ser feito ou o que pode ser concluído em situações diferentes. Um sistema baseado em regras é formado por um grupo de regras SE-ENTÃO, um grupo de fatos, e algum interpretador que controla a aplicação das regras. Podem ser vistas como uma simulação do comportamento cognitivo de especialistas humanos.

São também denominados *sistemas de produção* e é uma forma natural de representar o conhecimento. Exibem as seguintes características desejáveis:

- Modularidade: cada regra define uma pequena parte, relativamente independente, do conhecimento. Cada regra representa um "pedaço" de conhecimento independente.
- Transparência: é possível permitir a explicação das decisões e soluções.
- Incrementabilidade: novas regras podem ser adicionadas à base de conhecimento de forma relativamente independente das outras regras.
- Capacidade de serem modificadas: (conseqüência da modularidade) regras antigas podem ser modificadas relativamente independentemente de outras regras.

Há dois tipos de sistemas baseados em regras: sistemas de encadeamento para adiante (“forward chaining”), e sistemas de encadeamento para trás (“backward chaining”).

Em um sistema de encadeamento para diante (Figura 1. 10), também chamado dirigido a dados (“data driven”) começa-se com os fatos iniciais, e continua-se usando regras para obter daí novas conclusões (ou outras ações) determinada por esses fatos.

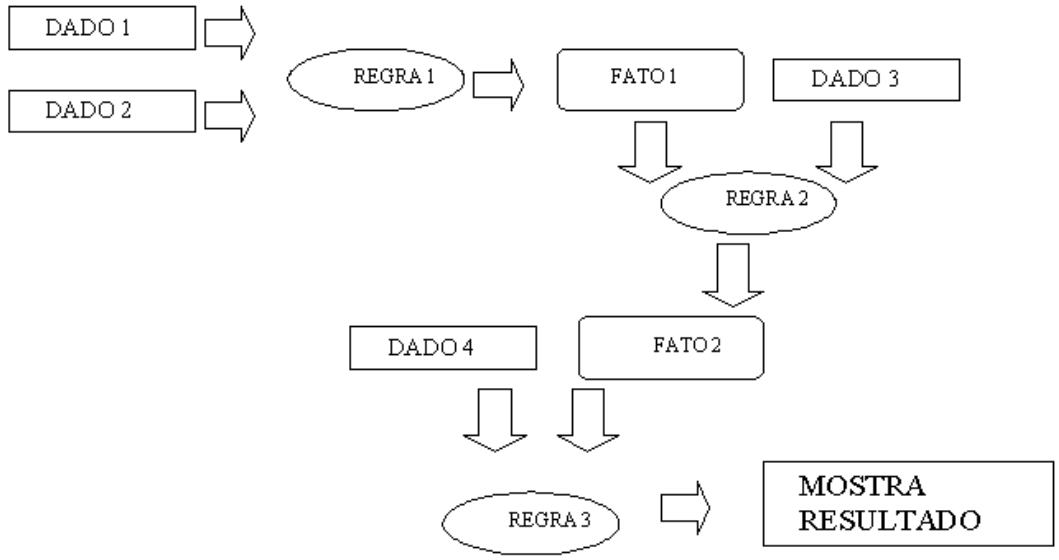


Figura 1. 10– Encadeamento para diante

Veja como o processo se sucede (Figura 1. 11):

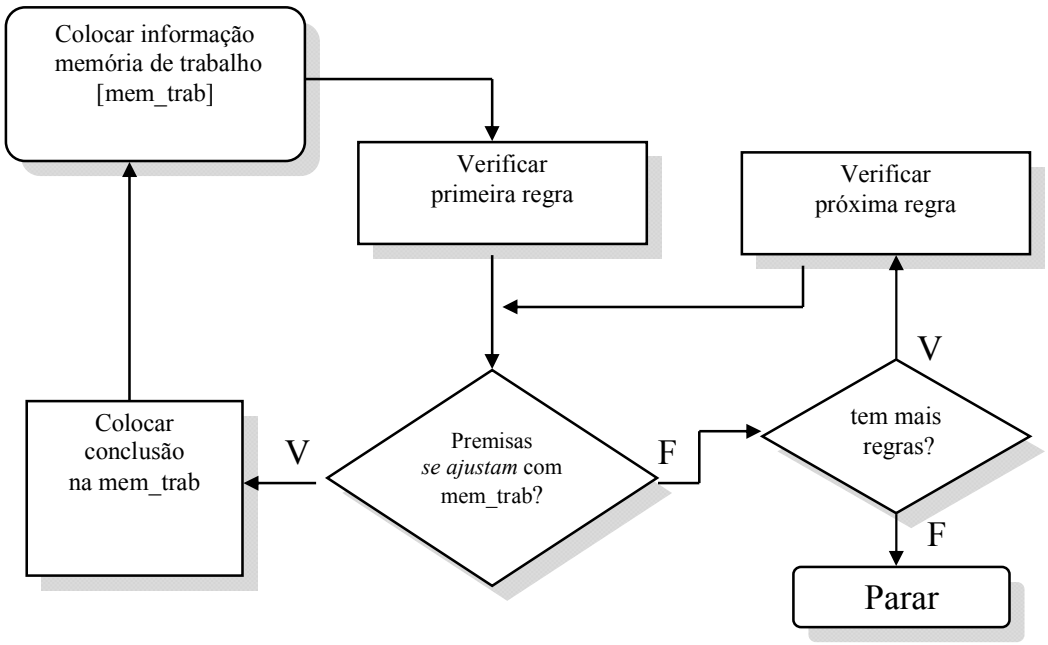


Figura 1. 11 -Encadeamento para frente

Em um sistema de encadeamento para trás (Figura 1.12) começa-se com alguma hipótese (ou meta) que se busca provar, e continua-se procurando regras que casem com a conclusão daquela hipótese. É o sistema empregado pelo Prolog.

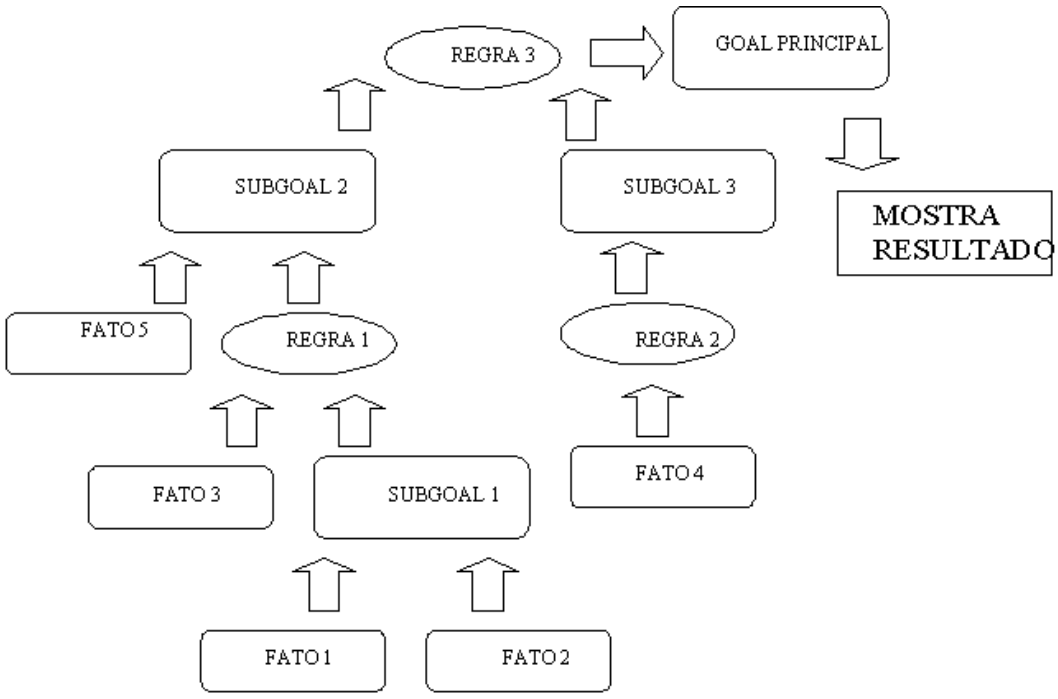


Figura 1. 12– Encadeamento para trás

Os Sistemas baseados em regras podem apresentar erros quando ocorre:

- Regras redundantes (a ordem das premissas é diferente e a conclusão é igual)
- Regras conflitantes (premissas iguais e conclusão diferente)
- Regras que têm a mesma conclusão, mas não as mesmas premissas.
- Regras circulares (premissa é igual à conclusão e conclusão é igual à premissa)
- Regras com premissas desnecessárias (a premissa é sempre satisfeita)
- Regras sem saída (não disparam outra regra)
- Regras perdidas (fatos não usados ou conclusões não afetadas por qualquer regra)
- Regras em que as premissas nunca são testadas (regras perdidas ou falta de dados na entrada)

Os sistemas de produção apresentam como componentes principais:

- Memória de trabalho: consiste em uma coleção de assertivas verdadeiras.
- Base de regras é o conjunto de sentenças (regras de inferência) que determinam as ações que devem ser tomadas de acordo com as percepções.
- Motor de inferência é a parte do sistema que determina o método de raciocínio. Utiliza estratégias de busca e resolve conflitos.

Do que já foi exposto, pode-se concluir que para a representação do conhecimento são utilizados: objetos estruturados, lógica, e regras.

Objetos estruturados são úteis para representar informação declarativa sobre coleções de relacionamento objetos/conceitos, e, em particular, onde há uma hierarquia de classe clara, e onde se quer usar *herança para deduzir os atributos de objetos em subdivisões de classe dos atributos de objetos na classe pai.*

Redes semânticas, inicialmente, eram pouco utilizadas, porém hodiernamente há alguns sistemas práticos que as utilizam na representação do conhecimento, possibilitando, com isto, semântica subjacente clara.

Objetos estruturados não são bons se o sistema tem como objetivo a obtenção de uma gama extensa de tipos diferentes de conclusões. Sistemas baseados em regra são mais adequados para este tipo de finalidade, porque permitem representação de fatos bastante complexa (envolvendo quantificação etc), e têm sintaxe e semântica bem definidas.

Sistemas baseados em regras tendem a permitir só representação relativamente simples dos fatos subjacentes dentro do domínio, mas podem ser mais flexíveis, e freqüen-

temente permitem certezas para avaliar fatos que são associados com regras (probabilidade da conclusão ser verdadeira).

Enquanto a lógica é principalmente usada de um modo declarativo, sistemas baseados em regras (especialmente sistemas de encadeamento para diante) estão mais preocupados com o conhecimento processual.

Alguns sistemas são construídos baseados em representação híbrida do conhecimento:

- Os "frames" podem ser referenciados por uma regra;
- As regras podem ser definidas como "frames", agrupadas em classes;
- Alguns "slots" de um "frame" podem conter regras.

1.4 Engenheiro do Conhecimento

É o profissional que extrai o conhecimento do(s) especialista(s) - ou de outra(s) fonte(s) – interpreta-o e representa-o em tipos e estruturas de conhecimento na base de conhecimento.

No o desempenho de sua função, são necessárias para qualificar um engenheiro do conhecimento as seguintes condições:

- Habilidade para aprender.
- Boa habilidade para comunicação.
- Habilidade de organização.
- Habilidade para conceitualizar.
- Habilidade de diplomacia.
- Habilidade de documentação.
- Conhecimento de técnicas de projeto de sistemas baseados no conhecimento.

Para o processo de aquisição de conhecimento, o engenheiro se vale de:

- Entrevista: É a técnica considerada mais comum e normalmente é empregada para elaboração de sistemas de bancos de conhecimento pequenos, requerendo, para isto, poucas horas de entrevista com o especialista, resultando num protótipo rápido de validação e verificação do conhecimento. É interessante que, também seja feito interação com o usuário que irá utilizar o sistema para verificação do que é desejado pelo mesmo.

- Rastreamento de processo: Uma série de técnicas que permite a determinação do processo de pensar de um indivíduo enquanto ele realiza uma tarefa ou chega a uma dada conclusão.
- Análise de protocolos: Constitui uma técnica para analisar os resultados de uma sessão de aquisição do conhecimento. Uma vez que a sessão de aquisição de conhecimento tenha sido conduzida, resultando num "histórico" registrado ou anotado, o engenheiro de conhecimento deve traduzi-lo em protocolos (registros gravados ou transcritos a partir da sessão) para análise posterior.
- "Brainstorming": Técnica de pesquisa de idéias originais, baseada na comunicação recíproca, num grupo, das associações livres de cada um de seus membros. Originou-se da preocupação de altos executivos com os gerentes que eles supervisionavam. Estes sempre repetiam e imitavam a sabedoria de seus superiores.
- Estudos de casos: Através de casos já documentados, elicita-se conhecimento do especialista. Uma desvantagem do método é a necessidade de haver material suficiente dentro do domínio para desenvolver o estudo. O sucesso do método depende do caso escolhido
- Introspecção: Especialista verbaliza sensações, memórias, sentimentos e imagens que percebe enquanto resolve um problema. Uma desvantagem do método é que o conhecimento talvez não reflita o processo de tomada de decisão usada pelo especialista, pois informações ficam no subconsciente.
- Repertório "grid": É uma técnica multidimensional que procura mostrar a estrutura de um conjunto particular de conceitos e identificar similaridades entre objetos agrupando-os conceitualmente. Normalmente é usado para obter informações sobre opiniões ou outros conhecimentos imprecisos.

Para o processo de aquisição do conhecimento, na maioria das vezes, segue-se o seguinte roteiro:

- Observação no local - observar o especialista resolver problemas no seu local de trabalho
- Discussão de problemas - tipos de dados, conhecimentos e procedimentos para resolver o problema.
- Descrição do problema - descrever um problema protótipo para cada categoria de pergunta do domínio (área)

- Análise do problema - Baseado no processo de resolução de uma série de problemas reais e o raciocínio empregado passo a passo.
- Refinamento do sistema - o especialista apresenta uma série de problemas para resolver usando as regras adquiridas das entrevistas
- Exame do sistema - o especialista examina e crítica o protótipo do sistema de regras e a estrutura de controle
- Validação - apresenta casos resolvidos pelo especialista e o protótipo do sistema a outros especialistas

Do ponto de vista comparativo com a computação tradicional pode-se comparar o trabalho do engenheiro do conhecimento como segue (Tabela 1. 4):

Engenharia do Conhecimento	***Programação Convencional***
Escolhe uma lógica	Escolhe uma linguagem de programação
Constrói uma Base de Conhecimento	Escreve um programa
Implementa uma prova teórica	Escolhe ou escreve um compilador
Infere novos fatos	Executa um programa
Respostas são derivadas da descrição do problema e da BC	Saída derivada a partir da entrada

Tabela 1. 4 – Comparação de Técnicas de Engenharia do Conhecimento X Programação Convencional

Portanto o Engenheiro do Conhecimento é:

- O profissional que extrai o conhecimento, interpreta-o e representa-o em estruturas de conhecimento, armazenando-os na base de conhecimentos.
- Aquele que deve se informar sobre o domínio do problema para poder iniciar o processo de extração.
- O responsável pela construção de um protótipo do sistema para validar o conhecimento previamente extraído.
- O responsável pela criação de um glossário de termos para facilitar o entendimento do domínio.
- O responsável pela manutenção do conhecimento, quando esta tarefa não puder ser realizada pelo especialista.

Capítulo 2
Sistemas Especialistas

2.1 Introdução

Um Sistema Especialista é formado por um conjunto de softwares que manuseia conhecimentos armazenados em bancos de dados (denominado base de conhecimento), e emprega técnicas de inferência com o objetivo de solucionar problemas, que até então só podiam ser resolvidos com a habilidade humana, tais como problemas não estruturados, nos quais é complicado um procedimento lógico para a sua resolução.

Os sistemas especialistas são elaborados com a finalidade de lidar com quase todos os tipos de situações em que são requeridos raciocínios formais para sua solução, como por exemplo: diagnósticos médicos, defeitos em equipamentos, previsões meteorológicas dentre outros [Harmon 1985] [Sabbatini A 1993] [Kojima at al 2000] .

O propósito destes sistemas não é o de substituir a o especialista humano, consistindo, na verdade, numa ferramenta útil para ampliação da sua experiência e conhecimentos. À medida que novas situações vão sendo identificadas, o acervo da base de conhecimento é realimentado, tornando as novas informações disponíveis a partir de sua introdução no sistema, incrementando, com isto, a produtividade e o aculturamento do utilizador.

O primeiro programa especialista baseado em conhecimento, chamado DENDRAL, foi escrito em 1967 [Buchanan at al 1978]. Este sistema usava uma representação procedural e era capaz de inferir estrutura molecular de componentes desconhecidos a partir do fornecimento da massa espectral e da resposta nuclear magnética destes elementos. Posteriormente, sistemas especialistas baseados em regras mais sofisticados foram desenvolvidos, notavelmente o programa MYCIN [Charniak at al 1985]. Ele utiliza regras derivadas do domínio médico (Infectologia) para raciocinar (deduzir) a partir de uma lista de sintomas de alguma doença em particular.

Em Singapura, desde 1980, os sistemas especialistas são utilizados amplamente em diversos setores especialmente na área bancária, financeira, na manufatura, dentre outros. Podem-se citar alguns dos importantes sistemas especialistas desenvolvidos e usados neste

País: Audit Expert System, no setor de contabilidade; Credit Evoluation, no setor bancário; Intelligent Fuzzy Logic Tutor, no setor de educação.

No Japão, um grande número de sistemas especialistas foi desenvolvido para diagnóstico, para planejamento, para escalonamento e para indústrias pesadas. Aplicações de sistemas especialistas associados à lógica difusa [Ebrahim 2001] estão se multiplicando, principalmente na área de eletrodomésticos.

Na Alemanha, os sistemas especialistas são utilizados sobretudo para indústrias pesadas e o uso de sistemas especialistas associados à lógica difusa está crescendo rapidamente.

Nos Estados Unidos da América, várias tendências são observadas no que concerne ao uso dos sistemas especialistas, como por exemplo:

- Um movimento contínuo em direção à integração e aos sistemas híbridos [Brasil at al 2001]; ênfase para o problema de "solução de negócios";
- Crescimento da tendência de sistemas de informação "ativos", ampla base de conhecimento, compartilhamento deste conhecimento e sistemas inteligentes híbridos;
- Necessidade de fornecer suporte de alto nível para pesquisa em Inteligência Artificial
- Uso de metodologias estruturadas para desenvolvimento de sistemas especialistas.

Dentre os inúmeros Sistemas Especialistas desenvolvidos, alguns são particularmente conhecidos devido a particularidades históricas [Firebaugh 1988] [Nilson 1980], pode-se citar:

- Mycin – Desenvolvido por Shortiliffe em 1973, na Universidade de Stanford, especialista em terapia antimicrobiana. O Mycin foi um pioneiro dentre os Sistemas especialistas e, para sua elaboração, foi consumido um esforço de aproximadamente 50 homens/ano, boa parte dele, embutido na elaboração da sua base de conhecimento. Diagnostica rapidamente meningite e outras infecções bacterianas, e prescreve seus tratamentos. Para a representação do conhecimento, utilizou-se uma estrutura baseada em regras probabilísticas (em torno de 500). Este sistema introduziu explicação e boa interface com usuário.
- Casnet – Desenvolvido por Szolovits e Weiss, em 1978. Especialista no diagnóstico e tratamento de Glaucoma.
- Puff – Desenvolvido em 1980, especialista em testes de função pulmonar.

- Prospector – Desenvolvido em 1978 por Gaschnig, especialista em exploração geológica.
- RI - especialista em configuração de computadores VAX-McDermott.
- Ace – Desenvolvido por Vesonder, em 1983. para manutenção de cabos telefônicos.
- SBDS (Service Bay Diagnostic System) desenvolvido pela Ford e a Hewlett Packard, em 1996, para diagnóstico de problemas em veículos automotores.

No Brasil, o Instituto Militar de Engenharia foi o pioneiro em pesquisas na área de Inteligência Artificial. Há alguns anos vem desenvolvendo várias ferramentas para aplicação das técnicas de Inteligência Artificial. Desenvolveu sistemas de recuperação em grandes bases de conhecimento, interpretadores de LISP e PROLOG, editores inteligentes, processos de jogos e interfaces para criação de ambientes em PROLOG. Seu objetivo principal é o desenvolvimento de sistemas especialistas e processadores de linguagem natural [Genaro 1986].

A Pontifícia Universidade Católica do Rio de Janeiro desenvolveu, na década de 1980, importantes trabalhos com sistemas especialistas. Seu principal produto é um sistema chamado SAFO cuja finalidade é a demonstração de teoremas matemáticos [Ribeiro 1983].

O Laboratório de Inteligência Artificial da Universidade Federal do Ceará merece destaque pelo desenvolvimento de um “shell” de sistema especialista – o Expert Sinta – desenvolvido no final da década de 90, utilizando uma linguagem orientada a objeto – o “object pascal” do Delphi – usado praticamente em todas as grandes Universidades brasileiras no ensino e desenvolvimento de sistemas inteligentes [Nogueira at al] [Silva at al].

2.1.1 Classificação

Os Sistemas Especialistas são empregados com duas finalidades básicas:

- Apoio à decisão: ajuda o "tomador de decisões" a lembrar-se de tópicos ou opções estratégicas do problema a ser resolvido;
- Tomada de decisão: toma a decisão em substituição a uma pessoa (uso mais comum).

Hayes-Roth oferece a seguinte tipologia de aplicações de sistemas especialistas [Hayes-Roth 1985]:

- Interpretação – são sistemas que inferem descrições de situações a partir da observação de fatos, isto é, fazem a análise de dados e procuram determinar as relações e seus significados.
- Predição - Deduzindo conseqüências prováveis de determinadas situações. A partir de uma modelagem de dados do passado e do presente este sistema permite uma determinada previsão do futuro.
- Diagnose - Deduzindo conclusões de fatos observados. São sistemas que interpretam falhas oriundas da interpretação de dados. A análise dessas falhas pode conduzir a uma conclusão diferente da simples interpretação de dados.
- Projeto - Configurando objetos debaixo de restrições.
- Planejamento – O sistema prepara um programa de iniciativas a serem tomadas para se atingir um determinado objetivo.
- Monitoramento - Comparando observações para planejar vulnerabilidade do sistema. Deve-se verificar, de maneira contínua, um determinado comportamento em limites pré-estabelecidos, sinalizando-se quando forem requeridas intervenções para o sucesso da execução.
- Depuração – Trata-se de sistemas que possuem mecanismos para fornecerem soluções para o mau funcionamento provocado por distorções de dados.
- Conserto – Desenvolve e executa planos para administrar os reparos feitos na fase de diagnóstico.
- Instrução - Diagnosticando, depurando e consertando comportamento de aprendizado dos estudantes [Villela 1993].
- Controle - Interpretando, predizendo, consertando e monitorando comportamentos de sistemas.

Sistemas Especialistas são utilizados, portanto, em problemas de gestão do tipo interpretação (de normas, de regulamento), diagnóstico (médico, análise financeira através de indicadores), previsão (escolha de investimentos, do tempo), planejamento, concepção (desenvolvidos a partir de projetos) e formação (suporte inteligente à instrução inteligente).

2.1.2 Benefícios

Os sistemas especialistas apresentam benefícios importantes notadamente no que se refere a:

- Criação de repositório de conhecimento
- Crescimento de produtividade e qualidade
- Habilidade de resolver problemas complexos
- Flexibilidade e modularidade
- Operação em ambientes arriscados
- Credibilidade
- Habilidade de trabalhar com informações incompletas ou incertas
- Fornecimento de treinamento

Na avaliação de Giarratano os Sistemas Especialistas apresentam características interessantes, que justificam seu uso e construção, quais sejam [Giarratano at al 1998]:

- Alta disponibilidade: Perícias estão disponíveis em qualquer "hardware", a qualquer momento, de modo satisfatório. O usuário empregará o sistema especialista para consulta sempre que desejar.
- Custo reduzido: O custo de prover destreza por usuário é reduzido.
- Perigo Reduzido: Sistemas especialistas podem ser usados em ambientes que poderiam ser perigosos para um humano.
- As perícias são permanentes: Peritos humanos podem se aposentar, se ausentar ou morrer. O conhecimento do sistema especialista durará indefinidamente.
- Perícias Múltiplas. Sistemas especialistas valem-se do conhecimento de múltiplos peritos estando disponível para trabalhar simultaneamente e continuamente qualquer hora do dia ou noite na resolução de um problema. O nível de conhecimento combinado de vários peritos pode exceder o de um único perito humano [Harmon 85].
- Aumento da confiabilidade: Sistemas especialistas aumentam a confiança em que a decisão correta foi tomada provendo uma segunda opinião para um perito humano ou serve de juiz no caso de haver discordâncias por parte de múltiplos peritos humanos. Claro que, este método provavelmente não terá valor se o sistema especialista foi programado por um dos peritos. O sistema especialista sempre deveria concordar com o perito, a menos que um engano fosse cometido pelo mesmo (isto pode acontecer se o perito humano estiver cansado ou submetido à tensão).
- Explicação: Sistema especialista explica com detalhes o raciocínio que conduziu a uma conclusão. Um perito humano pode estar muito cansado, pouco disposto ou

impossibilitado para fazer isto todo o tempo. Isto aumenta a confiança que a decisão correta foi tomada.

- Rapidez na resposta: Velocidade ou resposta em tempo real pode ser necessária para algumas aplicações e pode depender do "software" e "hardware" usados. Um sistema especialista pode responder mais rapidamente e pode estar mais disponível que um perito humano. Algumas situações de emergência podem requerer soluções mais rápidas que um humano e assim um sistema especialista, que forneça resposta em tempo real, é uma escolha boa [Hugh 88] [Ennis 86].
- Estabilidade: Resposta não emotiva e completa a toda hora pode ser muito importante, em tempo real e situações de emergência, quando um perito humano pode não operar com eficiência devido à tensão ou fadiga.
- Tutor inteligente: Sistema especialista pode agir como um tutor inteligente deixando o estudante rodar programas de amostra e fornecendo o raciocínio de funcionamento dos mesmos.
- Banco de dados Inteligente: Podem ser usados sistemas especialistas para se ter acesso a um banco de dados de uma maneira inteligente.

2.1.3 – Construção de sistemas especialistas

A construção de um sistema especialista envolve a extração do conhecimento pertinente a um ou mais peritos humanos. Tal conhecimento é freqüentemente heurístico em sua natureza, baseado em regras de produção em lugar de certezas absolutas. Um engenheiro do conhecimento tem o trabalho de extrair este conhecimento e construir a base de conhecimento do sistema especialista [Amaral at al 1993].

Aquisição de conhecimento para sistemas especialistas é uma área de intensa pesquisa, com uma variedade ampla de técnicas de aquisição.Geralmente assume importância no desenvolvimento inicial de um protótipo baseado em informação o engenheiro do conhecimento para extrair o conhecimento, entrevistando o perito e, a partir daí, iterativamente refinar esta base, baseada em avaliação que deverá ser feita pelo perito e por usuários potenciais do sistema especialista.

Para realização do processo de iteratividade do protótipo é importante que o sistema especialista seja escrito de tal modo que possa ser inspecionado facilmente e possa ser modificado. O sistema deve ser capaz de poder explicar todo o encadeamento (para o perito,

usuário e engenheiro de conhecimento) de perguntas e respostas relacionadas ao processo de solução de problemas.

O esquema de representação de conhecimento mais comumente usado para sistemas especialistas é o baseado em regras. Tipicamente, as regras não terão conclusões únicas - havendo algum grau de certeza associada a cada conclusão resultante do encadeamento de determinadas condições. Técnicas estatísticas determinam estes graus de certezas [Sakellaropoulos at al 2000].

São usados sistemas especialistas para resolver uma gama extensa de problemas em domínios como medicina, matemática, engenharia, geologia, informática, comércio, legislação, defesa e educação.

A construção de um "software" que tem como finalidade à geração de sistemas especialistas – denominados "shells" de sistemas especialistas - não é tarefa fácil, já que o programa deve ser apto à tarefa de tratar problemas complexos do mundo real que necessitam da interpretação de um perito, e concomitantemente devem ser capazes de chegarem às mesmas conclusões a que chegaria o especialista humano caso se defrontasse com problemas análogos.

A necessidade de utilização de ferramentas do tipo "shell" é motivada por diversos fatores tecnológicos e econômico-sociais, dentre os quais destaca-se a dificuldade de acesso a programadores especializados em Inteligência Artificial em determinadas regiões do planeta. Além disso:

- O "shell" possibilita o armazenamento e formalização do conhecimento de vários especialistas humanos
- Constitui ferramenta de apoio à tomada de decisões por parte do especialista; é fácil o treinamento de profissionais e, quando construída, possibilita imparcialidade na tomada de decisões.

No momento de implementação do sistema, com as regras já elaboradas, ao invés de termos sistema especialista enormes, também é possível a construção de vários sistemas integrados com outros, em redes corporativas, cada um desempenhando a sua função. Cada sistema é responsável por uma tarefa, são interligados através de redes, comunicando-se um com os outros para atingir uma determinada meta. Esse é o conceito de inteligência artificial distribuída.

2.2 Projeto de um sistema especialista

2.2.1 Escolha de um problema

Geralmente escrever um sistema especialista envolve um custo elevado. No processo de avaliação da relação custo / beneficio vários fatores são considerados [Causey 1994]:

1. *Peritos humanos não estão disponíveis em todas as situações onde são necessitados.* Se "o perito" do conhecimento é extensamente disponível, é improvável que valha a pena desenvolver um sistema especialista, porém, em áreas como exploração de óleo e medicina, onde o conhecimento especializado seja raro, poderá ser barato contar com sistemas especialistas para auxiliar tomadas de decisão.

2. *Problemas que fazem uso de técnicas de raciocínio simbólico poderão ser resolvidos facilmente por tais sistemas*. De um modo geral, estes problemas não requererem destreza manual ou habilidade física para serem resolvidos.

3. *Problema bem estruturado que não demanda (muito) conhecimento do bom senso.* O senso comum é notoriamente difícil de capturar e representar. Áreas altamente técnica são mais fáceis de representar e tende a envolver quantias relativamente pequenas de formalização do conhecimento.

4. *Os problemas não são facilmente resolvidos quando se usa método de computação mais tradicional.* Se há uma boa solução algorítmica para um problema, não é necessário usar um sistema especialista.

5. Existência de *peritos cooperativos e articulados.* Para um projeto de sistema especialista ter êxito é essencial que os peritos estejam dispostos a ajudar, e não sintam que exista ameaça ao seu emprego!

6. *O problema é de resolução trabalhosa.* Tipicamente elaborado para resolução de problemas que requeiram perícias altamente especializadas, mas que, no entanto, um perito humano levaria pouco tempo para resolver (diga-se uma hora no máximo).

7. Deve ficar claro que *só uma gama pequena de problemas é apropriada para usar tecnologia de sistema especialista, porém, determinado que a resolução para aquele problema é satisfatória, sistemas especialistas podem trazer benefícios enormes*. Sistemas foram, por exemplo, desenvolvidos para ajudar na análise de amostras colecionadas em exploração de óleo e ajuda de configuração de sistemas de computador. Ambos os sistemas estão, atualmente, em uso ativo, economizando quantias consideráveis de dinheiro.

Os sistemas especialistas também podem gerar problemas. Com a introdução da automação, os trabalhadores podem se sentir inseguros - eles podem achar que irão perder o emprego (e isto realmente pode acontecer). Esta situação pode ser evitada com uma migração lenta e responsável do sistema humano para o sistema máquina.

Um outro problema pode surgir quanto à política, sistemas especialistas não possuem senso de política, podendo produzir resultados embaraçosos. Portanto, em determinadas situações, é necessário subordinar o sistema a um analista humano que interagindo com o mesmo, quando necessário, desviará a atenção do fato embaraçoso.

2.2.2 Engenharia do conhecimento aplicada aos sistemas especialistas

Tendo decidido que o problema é adequado, precisa-se extrair o conhecimento do perito e representá-lo usando um "shell" de sistema especialista. O engenheiro do conhecimento atua como um intermediário entre o perito e o sistema especialista. O trabalho do engenheiro de conhecimento envolve a colaboração do(s) "expert"(s) e do(s) usuário(s) final (is).

O engenheiro do conhecimento é o perito em linguagem de representação em IA. Ele deve ser capaz de selecionar um "shell" de sistema especialista satisfatório (e outras ferramentas) para o projeto, extrair o conhecimento do perito, e implementar o conhecimento em uma base de conhecimento correta e eficiente.

Para extrair conhecimento do perito, o engenheiro de conhecimento tem que se tornar primeiro, pelo menos, um pouco familiar com o domínio de problema, talvez lendo textos introdutórios ou falando com o perito. Só após isto, a entrevista com o perito começa. Tipicamente, peritos são colocados diante de uma série de exemplos do problema e explicarão o raciocínio empregado para resolução dos mesmos em voz alta. O engenheiro do conhecimento elabora regras gerais abstratas destas explicações e das perguntas feitas ao perito.

Como ocorre com a maioria das aplicações, o sistema estará fadado ao fracasso se o usuário não ficar contente com ele. A fase de desenvolvimento deve envolver a colaboração íntima de usuários potenciais. No ciclo de desenvolvimento básico é de boa norma envolver o desenvolvimento de um protótipo inicial e iterativamente testá-lo e modificá-lo. Os peritos (conferem a validade das regras) e os usuários (conferem se o sistema fornece informação necessária) devem ficar satisfeitos com o desempenho e explicações do sistema.

Para desenvolver o protótipo inicial, o engenheiro de conhecimento tem que tomar decisões provisórias, encontrar representação de conhecimento apropriada e utilizar métodos a fim de obter conclusões (por exemplo, regras, ou regras+ "frames"; encadeamento para frente e/ou para trás). Para testar estas decisões de projeto básicas, o primeiro protótipo pode resolver só uma parte pequena do problema global. Se os métodos usados parecem trabalhar bem para aquela pequena parte vale a pena investir o esforço representando o resto do conhecimento da mesma forma.

A figura 2. 1 mostra esquematicamente as fases de desenvolvimento de um sistema especialista.

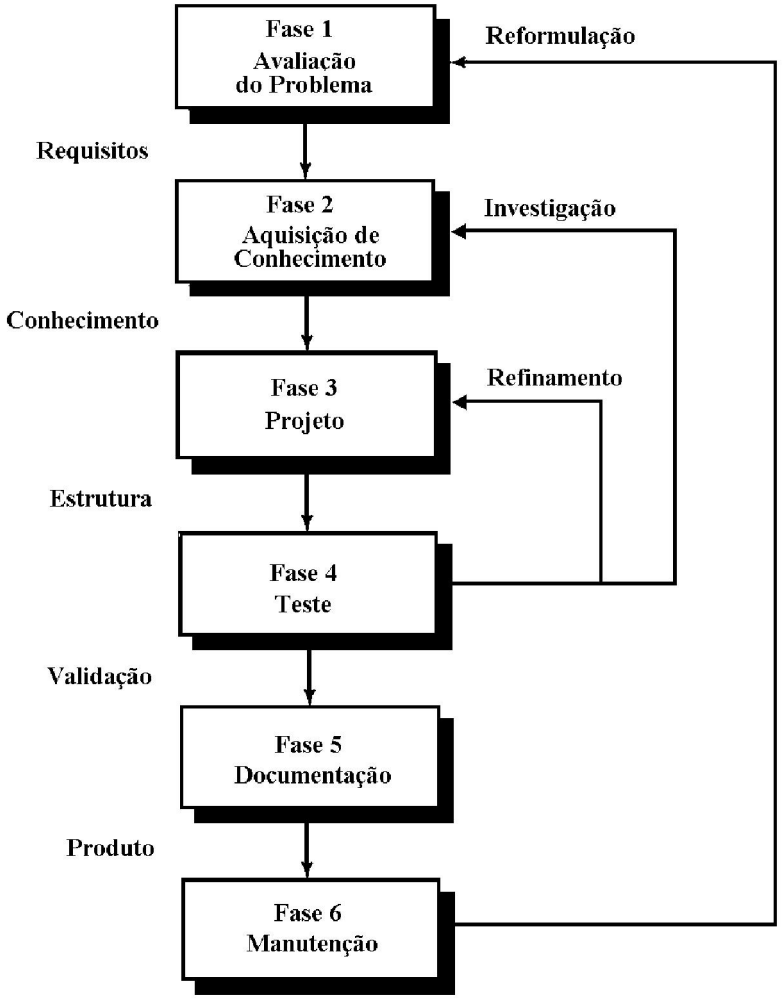


Figura 2. 1 – Fases de desenvolvimento de um sistema especialista

2.2.3 Regras e sistemas especialistas

Sistemas baseados em regra são do tipo meta dirigido quando usa encadeamento para trás, a fim de testar se alguma hipótese é verdadeira, ou são dados dirigidos, quando utilizam encadeamento para adiante para tirar novas conclusões de dados já existentes. O engenheiro do conhecimento pode usar qualquer uma das estratégias ou mesmo as duas. Uma razão para isto é que normalmente um sistema especialista terá que colecionar informação sobre o problema do usuário lhes fazendo perguntas - usar uma estratégia meta dirigida pode ser adequada se o sistema efetua perguntas ao usuário que são pertinentes a uma solução de hipótese.

De qualquer modo, um sistema especialista baseado em regras meta dirigido freqüentemente apresenta um conjunto de possíveis soluções para o problema – por exemplo, uma série de doenças que o paciente poderia ter. O sistema especialista pode considerar cada solução hipotética (por exemplo, Ter_febre(João) e, para este exemplo, tentar provar que a gripe é o resultado mais adequado para o caso. Às vezes, não é possível provar ou contestar algo a partir dos dados fornecidos pelo usuário, assim, o sistema fará para o usuário algumas perguntas (por exemplo, "Você tem dor de cabeça?"). Usando os dados iniciais e as respostas para estas perguntas, deverá ser capaz de concluir qual das prováveis soluções ao problema são as mais adequadas.

2.2.4 Tratamento da incerteza

As regras do sistema especialista podem ser vagas, trazendo insegurança ao usuário. Isto pode ser facilmente visualizado em um sistema de diagnóstico médico, em que o especialista não é capaz de ser definitivo no relacionamento entre os sintomas e as doenças. Na realidade, o médico oferece uma coleção de doenças possíveis. Isto se deve ao fato de certa incerteza ocorrer no processo de "inferência" da resposta por parte do médico [Abbod at al 2001].

Os sistemas especialistas que trabalham no mundo real apresentam, com certa freqüência, necessidade de manipular incertezas. Um esquema simples para manipulação de incerteza é associar um valor numérico a cada informação no sistema. O valor numérico representará o grau de certeza associado à informação. A combinação dos valores numéricos pode gerar o grau de certeza da resposta, pós-inferência.

Três abordagens são utilizadas para tratar a incerteza: lógica probabilística, fatores de certeza e probabilidade.

Lógica Probabilística

Também conhecida como lógica fuzzy [Castro at al 1999] [Ebrahim at 2001] [Fonseca at al 2001] [Hirano at al 2001] [Innocent at al 2001] [Lhata at al 2001]. A lógica fuzzy ou lógica difusa foi estruturada em 1965 por Lofti A. Zadeh, da Universidade da Califórnia, para tratar e representar incertezas. Fatos podem ser associados a números fracionários entre 0 e 1, conseguindo deste modo representar fatos que nem são totalmente verdadeiros e nem são totalmente falsos, conseguindo tratar a incerteza desta maneira.

A lógica difusa pode ser aplicada, por exemplo, na construção de sistemas especialistas para descrever coisas imprecisas como: altura (alto, baixo); velocidade (rápido, lento); tamanho (grande, médio, pequeno); quantidade (muito, razoável, pouco); idade (jovem, velho), dentre outros.

A lógica fuzzy objetiva fazer com que as decisões tomadas pela máquina se aproximem cada vez mais das decisões humanas, principalmente ao trabalhar com uma grande variedade de informações vagas e incertas, as quais podem ser traduzidas por expressões do tipo: a maioria, mais ou menos, talvez, etc.

Fatores de certeza

Neste tipo de abordagem, leva-se em consideração mais a prática do que a estatística propriamente dita. Os FC (Fatores de Confiabilidade) expressam o nível de confiabilidade que pode ser atribuído a uma conclusão.

Por exemplo, seja esta regra baseada no sistema Mycin:

SE a infecção é bacteriana

E o sítio de cultura é estéril

E a porta de entrada suspeita é o trato gastrintestinal

ENTÃO o organismo deve ser bacteróide (FC=0,7)

Isso seria a representação segundo a visão de um especialista.

Probabilidade

Probabilidade é o estudo de experimentos aleatórios ou não determinísticos. Nesta abordagem exigem-se números reais. Ou seja, deve ser feito um estudo estatístico em cima de cada evento e populações. Os programas médicos, por exemplo, construídos baseados no teorema de Bayes [McKendrick al 2000] [Sakellaropoulos at al 2000] usam como abordagem à construção de uma matriz contendo as probabilidades de um dado sintoma que pode ser associado a uma doença.

2.2.4.1 Redes de inferências

São grafos acíclicos dirigidos cujos nós representam fatos, podendo ser evidências ou conclusões.

O Grafo da Figura 2. 2 pode ser traduzido pela expressão: se (E1 $\wedge$ E2) $\vee$ ... En então Cx, onde x é a medida da incerteza da conclusão C.

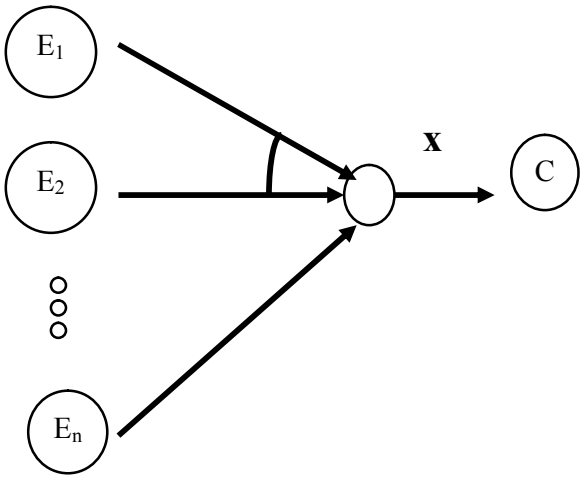


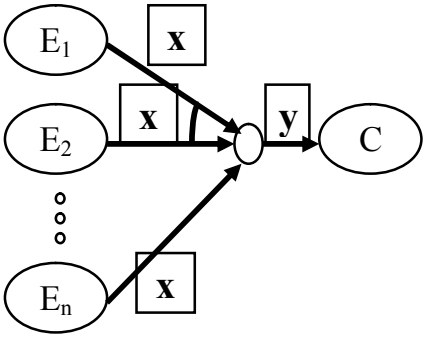
Figura 2. 2– Grafo e/ou

Redes Bayesianas, redes probabilísticas, redes causais e mapas de conhecimento são outros nomes conhecidos para o mesmo conceito.

Dentre os vários modelos, dois são bastante usados e clássicos:

- Modelo probabilístico: usa a regra de Bayes para propagar a incerteza.
- Modelo heurístico: baseado em regras heurísticas da lógica *Fuzzy* (normalmente os operadores max e min).

Há duas abordagens: a possibilística e a probabilístca dentro das regras heurísticas (baseada em Fatores de Certeza) [Rich at al 1994] [Giarratano at al 1998]. Na figura 2. 3 estas abordagens são evidenciadas.



	Possibilística	Probabilística
$E_1 \wedge E_2$	$\min(x_1,x_2)$	$x_1 \cdot x_2$
$E_1 \vee E_2$	$\max(x_1,x_2)$	$x_1 + x_2 - x_1.x_2$
$P(C) = P(x_1 \text{ op } x_2 \ldots \text{ op } x_n) \cdot y$		

Figura 2. 3– Grafos e/ou e sua abordagem possibilística e probabilística

2.3 Estrutura de um sistema especialista

A estrutura de um sistema especialista completamente operacional compreende cinco componentes essenciais:

2.3.1 Base de conhecimento

Local onde fica armazenada as informações, no nível de especialista, necessárias para resolução de problemas dentro de um domínio do conhecimento especifico.

O conhecimento de um sistema especialista é constituído por fatos e heurísticas. Os fatos correspondem às informações que estão disponibilizadas dentro do sistema para que possam ser compartilhadas e atualizadas pelo especialista no domínio do conhecimento. As heurísticas são regras representativas do processo de tomada de decisão de um especialista em um domínio.

O problema mais sério e complexo é a dificuldade de implementar um sistema que seja completo no domínio de conhecimento, sendo o tamanho e a qualidade da base de conhecimento fatores decisivos para determinação da eficiência e desempenho do sistema.

Uma base de conhecimento difere de um banco de dados convencional devido ao fato de incluir conhecimento explícito e conhecimento implícito. Muito do conhecimento na base não é declarado explicitamente, mas deduzido pela máquina de inferência a partir de declarações explícitas na base de conhecimento. Isto faz com que bases de conhecimen-

to tenham memória de dados mais eficiente que bancos de dados e lhes dá o poder para representar todo o conhecimento insinuado por declarações explícitas de conhecimento.

2.3.2 Interface de aquisição

Usada para alterar e adicionar conhecimento novo à base. É o componente utilizado pelo especialista e o engenheiro do conhecimento. Inclui características para ajudarem peritos a expressar seu conhecimento de uma forma satisfatória, interagindo com o computador.

2.3.3 Mecanismo de inferência

Também denominado interpretador de regras, corresponde à parte do programa que irá interagir com o usuário no modo de consulta e acessará a base de conhecimento para fazer inferências sobre o caso proposto pelo usuário. A máquina de inferência, de certa maneira, tenta imitar os tipos de pensamento que um especialista humano emprega para resolução de um determinado problema, ou seja, ele pode começar por uma conclusão e buscar uma evidência que a comprove; ou pode iniciar por uma evidência e a partir dela chegar a uma conclusão.

Como nem todo o sistema especialista utiliza a mesma abordagem de representação do conhecimento, o mecanismo de inferência deve ser projetado para adequasse ao formalismo especifico empregado.

O modo utilizado pelo motor de inferência para provar as regras da base de conhecimentos é denominado método de encadeamento e pode ser:

- Para Frente: “Forward Chaining” – “Data Driven” (a partir das premissas)
- Para Trás: “Backward Chaining” – “Goal Driven” (a partir das conclusões)
- Misto: com ações externas

Dependendo da maneira como o motor de inferência trabalha, pode-se classificá-lo em:

- Monotônico: retira fatos da memória de trabalho
- Não monotônico: não retira fatos da memória de trabalho
- Irrevogável: para quando o conjunto de conflitos = Ø (testas todas as possibilidades)
- Revogável: utiliza “backtraking”, desfaz para colher a primeira resposta.

  O sistema de busca utilizado pode ser em largura ou profundidade

2.3.4 Interface do usuário

A interface do usuário visa facilitar a comunicação entre o sistema especialista e o usuário. Esta interface é acionada cada vez que o usuário solicita uma explicação sobre uma decisão em particular que o sistema tomou, ou sobre qualquer fato ou conhecimento que ele guardou na base. A interface do usuário pode ser considerada boa quando reproduz o tipo de interação que seria esperada entre um perito humano e alguém consultando aquele perito.

2.3.5 Módulo de explicação

A maioria dos sistemas especialista tem módulos de explicação que permitem ao usuário obter a maneira que motivou o sistema a fazer alguma pergunta e como chegou a alguma conclusão. Estas perguntas são respondidas recorrendo às metas do sistema, as regras que são usadas e quaisquer dados pertinentes ao problema.

Prover tal facilidade de explicação envolve, pelo menos, o registro de quais regras são usados para obtenção das conclusões, e usar estes registros para compor explicações de como o sistema funciona.

A figura 2. 4 mostra esquematicamente o relacionamento entre os diversos componentes de um sistema especialista e dos seus utilizadores.

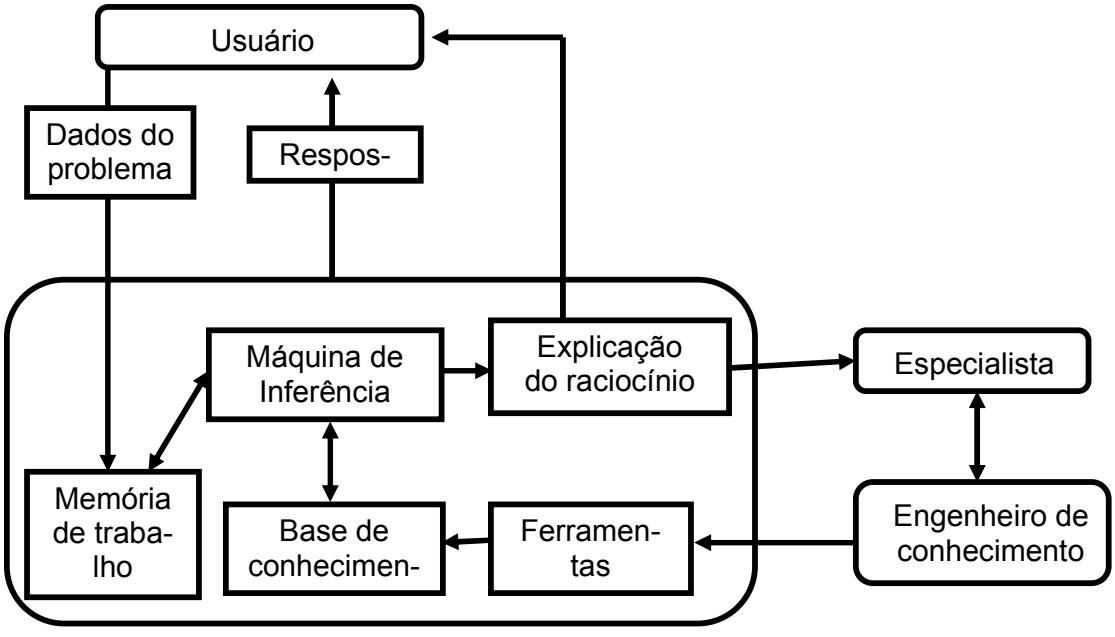


Figura 2. 4– Componentes de um sistema especialista

2.4 Ferramentas para construção de um sistema especialista

Para construção de sistemas especialistas, 3 opções são possíveis:

- "Shell" (OPS, Expert, KAS, Expert Sinta,etc...) : é o método mais utilizado
- Elementos "embutíveis": regras + objetos (CLIPS, JESS, NeOpus, JEPS, etc.)
- Linguagens de programação de alto nível (Prolog, Lisp, OOP).

Para definição da ferramenta mais adequada de construção de um sistema especialista são levados em consideração:

- Facilidade de uso
- Flexibilidade
- Interface com sistema
- Desempenho
- Portabilidade

A figura 2.5 esquematiza as opções de "softwares" disponíveis para construção de sistemas especialistas.

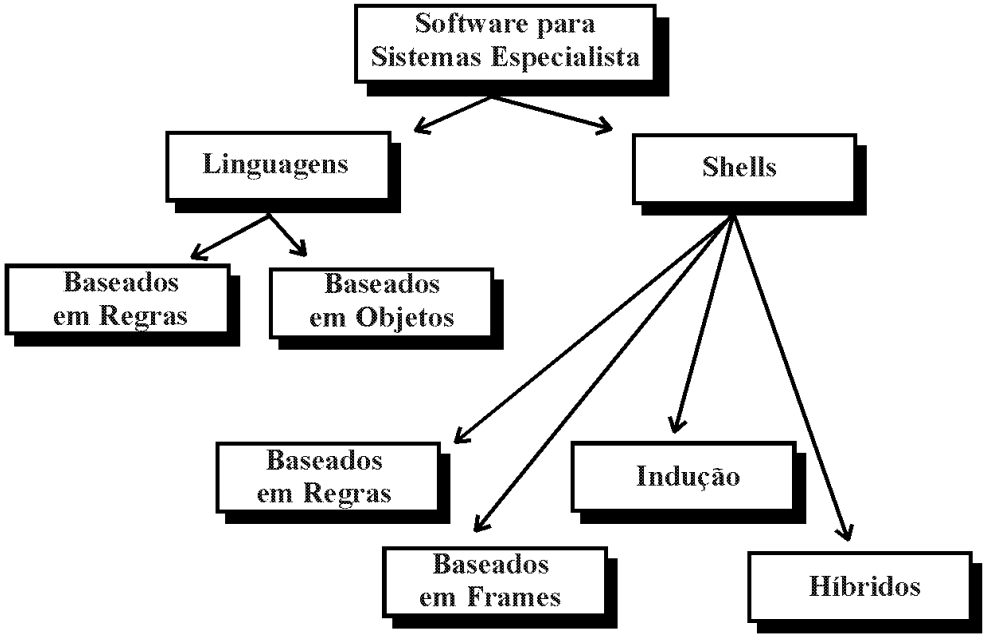


Figura 2. 5 – Categorias de "software" para construção de sistemas especialistas.

4.5 Personagens de um sistema especialista

Vários são as pessoas envolvidas com os sistemas especialistas desde a construção de ferramentas e linguagens de programação até os usuários finais do sistema. A figura 2.6 esquematiza a participação e relacionamento destes personagens:

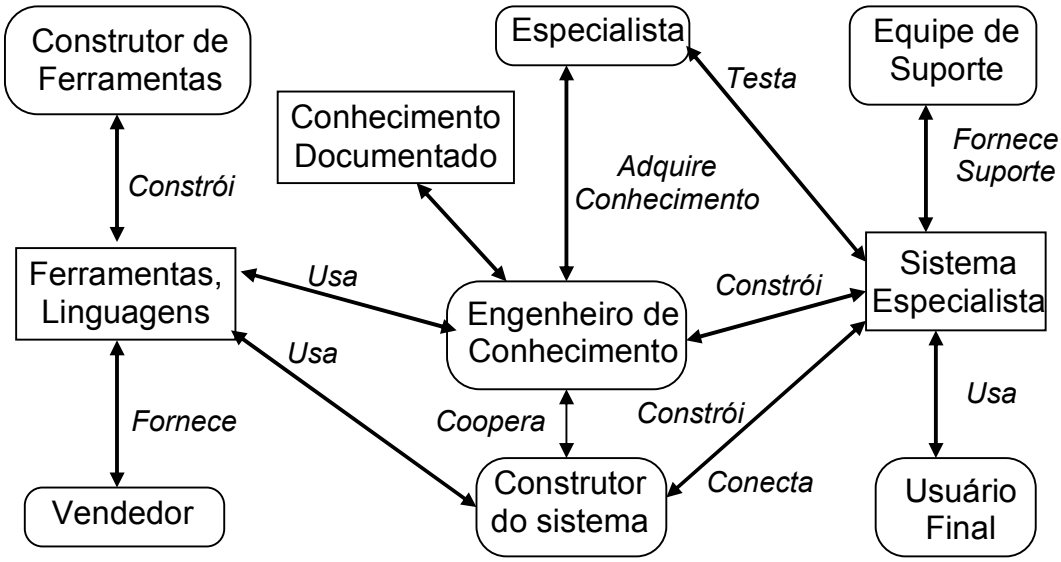


Figura 2. 6– Personagens de um sistema especialista

2.6 Sistemas especialistas na Medicina

O uso de computadores na pratica médica é atividade bastante difundida na atualidade, tendo destaque aplicações em áreas como: registro médico, processamento de imagens médicas (ultra-sonografia, tomografia computadorizada, angiografia digital, Medicina nuclear, etc.), processamento de sinais (E.C.G., E.E.G., miografia, etc.), sistemas de informação em saúde, registro médico, sistemas de apoio à decisão, etc.

No início, o emprego de computadores era restrito às grandes instituições hospitalares, isto devido ao alto custo dos equipamentos, e seu uso era quase sempre voltado para atividades administrativas. A partir do final da década de 70 e início dos anos 80, devido ao avanço tecnológico, ao barateamento dos equipamentos e principalmente ao aparecimento dos microcomputadores (PC) até mesmo departamentos de pequenos hospitais puderam adquirir e informatizar as rotinas de seus serviços.

O crescimento da informatização do ambiente hospitalar é enfatizado por Böhm. Para ele: “A perspectiva de uma presença maciça de computadores no dia a dia dos profis-

sionais de Saúde, em futuro próximo, e a consciência dos benefícios que a Informática pode trazer para a área tornam inevitáveis que nosso país venha a seguir o caminho que já está sendo trilhado pelas nações mais desenvolvidas, neste sentido" [Böhm 1994].

Embora o médico seja o principal, devido à especificidade do relacionamento médico – paciente, provedor de informações no ambiente hospitalar, ainda hoje se verifica certa rejeição do profissional da medicina ao uso mais generalizado do computador na prática clínica, isto se deve segundo Ilha [Ilha 1993] aos seguintes motivos:

- Abordagem mais administrativa do que médica
- Nenhuma contribuição para o processo médico ou suporte a terapia
- Inexistência de alívio da carga de trabalho do médico
- Pequena flexibilidade e falta de "amigabilidade"

A introdução de ambientes gráficos amigáveis (Windows, Macintosh, Linux), programas inteligentes que auxiliam no diagnostico e tratamento de pacientes, sistemas amigáveis e interativos que facilitam ao máximo o aprendizado do seu uso e, principalmente, o uso do computador na confecção do chamado prontuário eletrônico diminuiu consideravelmente esta rejeição tornando o computador, cada vez mais, componente obrigatório na rotina médica, ocorrência que se tornará mais relevante à medida que novos avanços tecnológicos forem alcançados, por exemplo, reconhecimento da voz.

2.6.1 Sistemas especialistas e Inteligência Artificial na Medicina

A partir da década de 70, um importante ramo da Informática, a Inteligência Artificial (definida como o ramo da Ciência da Computação que pesquisa metodologias para o desenvolvimento de sistemas que mimetizem o comportamento humano) começou a atuar, em caráter experimental, desenvolvendo programas denominados sistemas especialistas também conhecidos como sistemas baseados no conhecimento ou ainda "Expert System". Estes sistemas têm experimentado tremendo crescimento e popularidade desde que foi introduzido comercialmente no começo de 1980, e hoje sistemas especialistas são usados no comércio, ciência, engenharia, industria e muitos outros campos.

Professor Edward Feigenbaum, da Universidade de Stanford, um pioneiro na tecnologia de sistemas especialistas, define sistema especialista como "*... um programa inteligente de computador que usa conhecimento e procedimento de inferência que são difíceis o suficiente para requerer significativa perícia humana para sua solução*".Giarratano complementa "*...um sistema especialista é um sistema de computador que emula a habili-*

dade da tomada de decisão de um especialista humano. O termo emular significa que o sistema tem a intenção de atuar em todos os seus aspectos tal qual um especialista." [Giarratano 1988]

Se desempenho fosse o critério exclusivo para etiquetar um programa como sistema especialista, muitos sistemas de apoio de decisão, programas estatísticos e planilhas eletrônicas poderiam ser denominados como tal. Ao invés, o termo "sistema especialista" é normalmente reservado para programas de computador que alcançam desempenho de perito em um domínio substantivo específico que usa técnicas de inteligência artificial como representação simbólica, conclusão, e procedimentos heurísticos [Buchanan 1978].

Os sistemas de inteligência artificial médica surgiram para dar apoio aos profissionais da saúde em suas atividades rotineiras, auxiliando em atividades como manipulação de dados e aquisição de conhecimento. Em 1984, Clancey e Shortliffe deram a seguinte definição à inteligência artificial aplicada a Medicina: "Inteligência Artificial médica se preocupa primariamente com a construção de programas de IA que realizam diagnósticos e fazem recomendações terapêuticas" [Shortliffe 1990]. Os programas de IA aplicados à medicina são baseados em modelos simbólicos das entidades nosológicas e suas relações com os fatores ligados ao paciente e às manifestações clínicas. Em decorrência de mudanças resultantes do aperfeiçoamento dos sistemas especialistas esta definição seria considerada atualmente como limitada, em abrangência e visão.

Porque o poder de sistemas especialistas deriva da base de conhecimento em um domínio específico, sistemas especialistas, como peritos humanos, só são "especialistas" em áreas específicas. Na área médica houve proliferação de sistemas especialistas porque é mais fácil implementar bases de conhecimento médico mais restritas. "*O desenvolvimento de programas específicos para a área médica se torna mais fácil graças ao surgimento de profissionais híbridos: os informatas médicos ou médicos informatas, que associam conhecimento de informática a conhecimentos específicos da área de saúde"*.[Volpe at al 1994].

2.6.2 O Diagnóstico médico e o emprego de sistemas especialistas

O diagnóstico de doenças é a área mais complexa e difícil na tomada de decisão do médico, as razões são as que se seguem na avaliação de Sabbatini [Sabbatini [A] 1993]:

- O diagnóstico médico depende da análise de dados e informações de diversas fontes de natureza muito diferentes, incluindo a experiência prévia do médico em rea-

lizar diagnóstico do mesmo tipo, bem como o senso comum e a intuição. É bastante difícil formalizar este conhecimento e representá-lo através de um programa de computador;

- Os mecanismos mentais e o processo de raciocínio pelo qual o clínico chega ao diagnóstico é ainda mal conhecido. Ele envolve, simultaneamente, processos lógicos, avaliação probabilística, encadeamento causal e muitos outros processos ainda entendidos parcialmente. É difícil, portanto, simular num computador um modelo completo de raciocínio;
- Existe uma falta de padronização quanto aos termos e definições médicas, apesar de alguns esforços nesta área (CID, SNOMED, LPM, entre outros). Muitas vezes não existem opiniões consensuais por parte dos especialistas na área de estudo sobre como decidir em face de evidências conflitantes. São raros os bancos de dados confiáveis, e o conhecimento está espalhado em muitas publicações, lugares e cabeças.

Na Medicina, a partir da utilização de algoritmos que empregam raciocínio dedutivo semelhante aquele utilizado pelo médico em sua prática diária, é possível a identificação e tratamento de problemas na área de saúde. Fornecendo-se sintomas e sinais bem como os resultados dos exames complementares de um determinado caso clínico pode-se obter, com razoável precisão, os prováveis diagnósticos, bem como as razões e o raciocínio utilizados para chegarem até ele (alguns programas conseguem produzir referências bibliográficas). Esses raciocínios automatizados utilizam uma base de conhecimento sobre a especialidade médica em questão, armazenada previamente no computador. " *... Afinal, qual o profissional que não gostaria de poder digitar em um computador os sinais e sintomas encontrados em seu paciente, e obter imediatamente um elenco das hipóteses diagnosticas mais prováveis ?"* [Palombo at al 1993].

Volpe e Sabattini argumentam *" ...uma deficiência do processo hipotético - dedutivo é que o cérebro humano consegue trabalhar simultaneamente com um número muito restrito de hipóteses diagnosticas (pouco mais que três ou quatro). Além disso, nem sempre os dados decisivos são expostos e levados em consideração, principalmente pelos profissionais sem experiência, e isto pode conduzir a hipóteses altamente improváveis em face de um diagnóstico óbvio."* Para em seguida concluir *" Uma das vantagens do uso do computador no apoio ao diagnóstico médico é que os chamados sistemas especialistas (que utilizam técnica de inteligência artificial) tratam este problema fundamentados em uma*

série de nodos (sintomas, achados laboratoriais e achados de exames físicos) ligados por padrões específicos, para um dado diagnóstico, de forma que podem trabalhar com muitas hipóteses diagnosticas ao mesmo tempo." [Volpe at al 1994]

Os empregos de sistemas baseados no conhecimento são importantes notadamente no que se refere ao processo de tomada de decisão para realização de procedimentos médicos. Segundo Peter Pritchard [Pritchard 1995] isto se deve ao fato que estes programas:

- Dão acesso oportuno a conhecimento pertinente.
- Provêem diretrizes fáceis de usar para alcance grande de tarefas paciente-específicas.
- Ajudam o médico a ordenar as opções de decisão.
- Recordam o doutor de perigos potenciais.
- Avaliam o processo de conformar a diretrizes (concorrentemente) .
- Avaliam os resultados (retroativamente).

2.6.3 Estrutura e funções de um sistema especialista médico

As principais funções e características dos sistemas especialistas são [Falsarela, Chaves]:

- Armazenar o conhecimento e as experiências de especialistas em bases de conhecimento.
- Utilizar mecanismos de inferência integrados em bases de conhecimento para resolver- ou auxiliar a resolver - problemas.
- Possibilitar a inclusão de novos conhecimentos na base de conhecimento sem eliminar os conhecimentos já armazenados.

A estrutura interna de um sistema especialista é constituída por três partes: A base de conhecimento, o banco de dados e o interpretador de regras.

A base de conhecimento é formada por um conjunto de regras de inferências que são utilizadas no raciocínio para tomada de uma determinada decisão. A maioria dos sistemas adota regras baseadas na informação *se ... então* para representar o conhecimento. Os sistemas podem ter desde dezenas até milhares de regras. A interface de aquisição do conhecimento controla como o perito e engenheiro de conhecimento interagem com o programa para incorporar conhecimento na base de conhecimento.

O banco de dados fornece o contexto do problema dominante e é considerado como um conjunto de fatos úteis. Estes fatos são aqueles que satisfazem as regras que levam a conclusão de um determinado fato (diagnóstico, terapêutica, prognóstico, etc.).

O interpretador de regras é uma máquina de inferência e controla a base de conhecimento usando um conjunto de fatos para a produção de mais fatos (A partir da alimentação do sistema com dados específicos obtemos informações úteis na tomada de decisão).

A interface do usuário é a parte do programa que interage com o usuário. Permite ao usuário fornecer informações a fim de que haja resolução de problemas, exibe conclusões e explica seu funcionamento (raciocínio utilizado na resolução do problema).

A comunicação com o sistema deve ser feita idealmente empregando interface que faça uso de linguagem natural e ambiente preferencialmente gráfico, pois isto permite que o usuário faça uso do sistema de maneira quase intuitiva, sem a necessidade do auxilio do especialista (aquele que alimenta a base de conhecimento).

2.6.4 Sistemas especialistas na área médica mais conhecidos

Estima-se que mais de 2000 programas foram desenvolvidos fornecendo suporte à tomada de decisão, muitos deles incorporados a equipamentos biomédicos e a sistemas de uso rotineiro em hospitais e clínicas.

A principio, os programas desenvolvidos na área de Inteligência Artificial para apoio a decisão eram especialmente desenvolvidos para este fim. Com a evolução, apareceram programas denominados "*Shell" de Sistema Especialista* que possibilitam a confecção de sistemas que trabalham com diferentes bases de conhecimento usando um mesmo programa, diminuindo conseqüentemente o custo de implementação (Emycin, Clips, Niacin, VP Expert, Expertmd, Expert Sinta, InstantTea entre outros).

Os primeiros trabalhos em sistemas especialistas aplicados à Medicina foram produzidos em diversas universidades, a exemplo do MIT, Tufts University, Pittsburgh, Stanford e Rutgers, conduzidos por pesquisadores como Peter Szolovits, Edward Shortliffe e Randolph Miller. O campo atraiu muito dos melhores cientistas na área de Inteligência Artificial e a produtividade desta primeira década de trabalho continua sendo até hoje uma notável realização.

O primeiro artigo da literatura a aventar a possibilidade do uso de computadores como ferramentas de auxílio ao diagnóstico de doenças surgiu nos fins dos anos 50. Poucos anos depois, no inicio da década de 60, alguns protótipos mostraram-se eficientes. A

partir dos anos 70, começou a ser utilizada técnicas de Inteligência Artificial em Medicina, aparecendo um grande número de sistemas de apoio à decisão, de média ou grande complexidade, alguns exemplos são citados a seguir [Szolovits 1982] [Sabbatini 1993 [A]] [Widman 1998] [Coieira 1998] [Amaral 1995, 1996]:

MYCIN (1972-80): Programa interativo, desenvolvido nos Estados Unidos por Shortliffe e colaboradores, para diagnóstico de doenças infecciosas e prescrição de terapêutica antimicrobiana, podendo fornecer ao usuário explicação sobre o raciocínio utilizado em detalhes. O MYCIN foi escrito num sistema geral de produção chamado EMYCIN (Empty MYCIN), também chamado sistema de produção ou "SHELL" (Um sistema de produção é estruturado a partir de uma coleção de regras). O MYCIN estendeu a noção de que a base de conhecimentos deveria ser separada da maquina de inferências. Embora tenha sido desenvolvido para fornecer consultas aos clínicos, possuía a habilidade de explicar ambas as linhas: de raciocínio e de seus conhecimentos. Devido ao ritmo de desenvolvimento da Medicina, a base de conhecimentos foi elaborada para permitir atualização através da alimentação de dados pelo especialista. Uma limitação do seu uso foi à dificuldade de implementar condições para que aprendesse a partir de sua experiência. Embora não tenha sido utilizado amplamente pelos clínicos, forneceu importantes subsídios para o desenvolvimento de pesquisas na área de Inteligência Artificial.

DE DOMBAL'S SYSTEM (1972): Desenvolvido por Dombal e colaboradores a partir do final dos anos 60 na Universidade Leeds. O Sistema utiliza dados de sensibilidade, especificidade e prevalência da doença para vários sinais , sintomas e resultados de exames laboratoriais, empregando como mecanismo de análise estatística o obtido a partir do Teorema de Bayes, fornecendo a probabilidade para 7 possíveis causas de dor abdominal: apendicite, diverticulite, úlcera péptica perfurada, colecistite, obstrução do intestino delgado, pancreatite e dor abdominal inespecífica. O uso rotineiro deste sistema em hospitais da Inglaterra reduziu a taxa de incidência em 50% das apendicites perfuradas e diminuiu a quantidade de cirurgias abdominais desnecessárias de 36% para 14%.

CADUCEUS/INTERNIST: Sistema geral de diagnóstico assistido por computador aplicado na área de ensino médico. Um elemento importante implementado neste sistema é a utilização, de forma semelhante ao raciocínio empregado pelo médico, do encadeamento causal. Os médicos normalmente sabem porque um determinado sintoma ou resultado de teste aparece, como conseqüência de alguma alteração anatômica, bioquímica ou fisiológica, e isto é amplamente utilizado neste sistema.

DTA (1978): Sistema elaborado para acompanhamento de paciente em uso de Digital.

PIP : Sistema de diagnóstico em Medicina interna e Nefrologia.

CASNET / GLAUCOMA (1982): Diagnóstico e orientação terapêutica nos casos de glaucoma. Também faz uso de redes causais.

PUFF (1977-79): Primeiro sistema construído utilizando como ferramenta o EMECYN (Permite construção de sistemas especialistas baseados em regra). O PUFF diagnostica a presença e severidade de doenças pulmonares e produz relato para o arquivo do paciente. Existe versão para uso comercial.

ABEL (1982): Permite a identificação de distúrbios hidroeletrolitícos e metabólicos (ácido - básico) e faz aconselhamento terapêutico.

VM (1977 - 1981): o VM (Ventilator Maneger) interpreta dados quantitativos e qualitativos em unidades de tratamento intensivo e aconselha o intensivista no manuseio da mecânica ventilatória do paciente em pós-operatório. Alguns conceitos do programa foram implementados internamente em dispositivos de monitoramento ventilatório.

ONCOCIN (1989): Seleção de protocolos na terapia do câncer.

HELP: Introduzido por Warner em 1979 e aperfeiçoado por Pryor e colaboradores em 1983. Gera relato de conselhos baseados na combinação de dados paciente - especifico contidos em seu banco de dados.

AI-RHEUM : diagnose e tratamento de desordens reumatológicas.

RMRS (1984): Desenvolvido por McDonald e colaboradores. Monitoriza registros clínicos de pacientes armazenados "on line" e envia indicativos clínicos que deverão ser lembrados ao médico (Datas de vacinas, exames laboratoriais que precisam ser checados, etc.).

ATTENDING: (1986) desenvolvido pelo Prof. Sabbatini, apresenta procedimento para decisão de várias condutas médicas (diagnóstico, avaliação de risco, protocolo terapêutico, interpretação de alguns testes laboratoriais). Desenvolvido na Unicamp - Brasil.

ILLIAD: desenvolvido na Universidade de Utah, em Salt Lake City. Diagnóstico na área de Medicina Interna a partir de uma base de conhecimento que contem 908 doenças, 1.509 síndromes e 11.900 sinais, sintomas e exames subsidiários. O sistema é baseado em uma abordagem probabilística de apoio a decisão (Teorema de Bayes) fornece orientação terapêutica, depois de realizar o diagnóstico. "*O ILIAD oferece ainda um módulo especialista, que o aluno pode consultar como se fosse um preceptor médico, verificando assim a*

conduta apropriada de um especialista ante a situação proposta pelo aluno, e um módulo tutorial, em que o conhecimento a respeito de uma determinada patologia é diretamente transmitido" [Vilela 1993].

QUICK MEDICAL REFERENCE: Apoio ao diagnóstico em Medicina Interna, um dos sistemas comerciais mais utilizados no mundo.

HEART ATTACK SURVIVAL CALCULATOR: Primeiro arquétipo de sistema distribuído na Internet na área de apoio a decisão médica. O programa aceita valores clínicos, a exemplo do grau de oclusão da artéria coronariana conhecida e estima a probabilidade de sobrevivência do paciente ainda durante a fase de internação.

DIABOLO: desenvolvido pela empresa Cognitech, da França, primeiro sistema especialista de grande porte concebido para o paciente com o objetivo de auxiliar a terapia e a educação do diabético insulino - dependente. Este sistema apresenta dois mecanismos integrados de inferência: o primeiro ativa uma base de conhecimento de cerca de 300 regras é um especialista médico em Diabetologia e o segundo com cerca de 40 regras faz um levantamento do perfil individual do paciente, seu histórico nosológico e terapêutico e sua interação pedagógica com o sistema.

DSP (Sistema de Diagnóstico em Psiquiatria): Desenvolvido por M. B. do Amaral e colaboradores em 1995 no Hospital Universitário, Chiba, Japão. Composto por 1508 regras relacionando 208 achados clínicos com 257 diagnósticos. O sistema inclui os 30 grupos de diagnósticos psiquiátricos que são classificados baseados nas categorias 290 a 319 do DSM-III-R e o CID 9.

SEC: O Projeto SEC foi concebido junto com a Fundação Bahiana de Cardiologia - FBC, com objetivo de apoiar o médico não cardiologista no diagnóstico de eventos agudos da cardiopatia isquêmica. Seus usuários são, portanto, médicos não especialistas em cardiologia em unidades periféricas urbanas de atendimento, de natureza primária da rede pública de saúde. O SEC pode, ainda, ser utilizado por estudantes de medicina para fins de aprendizado, sem, no entanto, ter as características de um tutorial.

DXPLAIN: Desenvolvido por Octo Barnett no Massachussets General Hospital. Trata-se de programa completo de apoio a decisão médica distribuído na Internet para pessoas cadastradas. O Sistema contém uma base de possibilidades para cerca de 4,5 mil manifestações clínicas associadas a 2 mil doenças diferentes.

Capítulo 3

Sistema Gerador de Regras a partir de grafos valorados

3.1 Definição de grafo

Grafo é um conjunto finito ou infinito de nodos conectados por arcos. Se os arcos vão de um elemento a outro, em um sentido, tem-se um grafo dirigido. Se um arco vai de um nodo a1 a outro a2, diz-se que a2 é sucessor de a1 e a1 é antecessor de a2.

Uma árvore é um caso particular de grafo no qual para cada nodo existe no máximo um como antecessor.

A um nodo que não apresenta antecessor se denomina raiz ou fonte. Os nodos sem sucessores são chamados de ponta, folhas ou sumidouros. Os nodos raízes, por definição, têm profundidade zero. A profundidade de um nodo é a de seu antecessor mais um.

Em inteligência artificial, é habitual reportar o processo de busca através de grafos onde:

- Nó - Um ponto discreto e possivelmente uma solução.
- Nó filho - Nó ligado a outro nó em um nível anterior.
- Nó terminal - Um nó que finaliza um caminho.
- Espaço de busca - O conjunto de todos os nós.
- Objetivo ("Goal") - O nó que é o objeto da busca.
- Caminho da solução - Um grafo direcionado dos nós visitados que levaram à solução.
- Retrocesso ("Backtracking") - Retorno a um nó pai após a falha na pesquisa a partir do nó corrente.

Um tipo particular de grafo também utilizado em IA é o "grafo e/ou" que é um hipergrafo formado por hiperarco onde cada nodo pode ter dois tipos de sucessores:

- Nodos e: Temos que satisfazer a todos os seus sucessores para que o nodo se satisfaça.
- Nodos ou: Basta satisfazer a um sucessor para que o nodo se satisfaça.

3.2 Objetivo

Um dos esquemas de representação do conhecimento médico mais amplamente utilizada em livros textos é o fluxograma de decisão. A navegação nestes fluxogramas permite que se possa concluir por diagnósticos prováveis, melhor conduta para o paciente, que exames deverão ser solicitados, tipos de tratamentos, dentre outras coisas.

Um fluxograma é conceitualmente a ferramenta mais simples para representar a tomada de decisão. Codifica, em princípio, as sucessões de ações que um bom clínico executaria para qualquer paciente de qualquer população. Por exemplo, pode-se representar todas as sucessões de perguntas elaboradas, respostas dadas, que procedimentos poderão ser executados, análises de exames laboratoriais obtidos e diagnósticos eventuais, tratamentos e resultados para vários pacientes que apresentam no serviço de emergência com *hemorragia digestiva* ou qualquer outra patologia específica.

Se for observado uma quantidade grande de pacientes e permitir-se que especialistas, munidos ou não de ferramentas (por exemplo, "Data Mining") possam realizar uma análise retrospectiva apropriada de cada caso baseado no conhecimento na área de análise, pode-se identificar uma sucessão satisfatória de ações comuns para levar frente a todas possíveis circunstâncias que possibilitam solucionar uma dada tarefa. Muitos programas na área de sistemas especialistas foram elaborados empregando esta abordagem (Ex.: ABEL [Szolovits 1982]).

É possível implementar fluxogramas de decisão utilizando como estrutura de dados os grafos valorados. Grafo valorado é aquele no qual as arestas (arcos) apresentam valores. Um nodo origem poderia ter vários arcos, cada um representando um valor. O conjunto de todos os nodos com seus arcos, desde a origem até todos os nós terminais, representaria todos os espaços de buscas e cada nodo sumidouro seria uma possível solução.

Na medida em que um dado fluxograma representa o conhecimento médico para tomada de decisão em uma dada situação clínica e todos os possíveis espaços de buscas podem ser representados como regras de produção, por que não construir pequenos módulos de sistemas especialistas, baseados nestes fluxogramas, para que o aluno do curso de Medicina, podendo interagir com o sistema, possa simular atendimento a um paciente real?

Com o objetivo de gerar os fatos e elaborar regras para alimentar um "shell" de sistema especialista baseado em regras, foi desenvolvido um programa que, baseado no desenho de um fluxograma de decisão (representação do conhecimento médico em determinada

área especifica), torna a tarefa de construção de sistemas especialistas factível até mesmo para estudantes que tenham conhecimentos superficiais de técnicas de inteligência artificial.

3.3 Metodologia

O programa 'Gerador de Regras para Sistemas Especialistas' (SGR) foi desenvolvido para ser uma ferramenta auxiliar na empreitada de construção de sistemas especialistas, tendo sido utilizado na tarefa de sua construção a linguagem de programação Object Pascal (Visual Borland Delphi). O material do trabalho divide-se em duas partes: a primeira é a documentação estrutural e operacional que contém as descrições formais de todas as estruturas dos principais algoritmos utilizados, apresentados neste capítulo, bem como o código fonte comentado (Veja o Anexo II para o código completo). A segunda é o programa propriamente dito (Veja o ambiente de desenvolvimento do SGR no capitulo 4). Será descrito também o que cada estrutura representa dentro de um sistema especialista.

A obtenção dos fatos e das heurísticas (conjunto de regras) que permitem a confecção de um sistema especialista são obtidos a partir da elaboração de grafos valorados em uma área de desenho na tela de computador (Objeto TPaintBox do Delphi) baseados em fluxogramas de decisão para uma determinada tarefa na área médica. Os fluxogramas poderão ser confeccionados baseados no que professor/aluno considera mais adequado para representar um caso clínico particular ou a partir de livros textos que contenham este tipo de representação.

Para que cada nodo componente do grafo seja introduzido no TpaintBox é necessário o preenchimento de um formulário contendo as propriedades deste nó.

Cada nó apresenta propriedades (somente são referenciadas as propriedades utilizadas no preenchimento do formulário que serão aproveitadas para implementação do sistema especialista - para informação mais completa de todos os elementos constituintes dos nodos veja tabelas mais adiante), dispostas como numa tabela relacional com as seguintes informações:

- Código do nodo: No 'Sistema Gerador de Regras' é apenas um índice para individualizar cada nodo. Não será usado no sistema especialista. Serve também para simplificar o desenho do grafo, e quando da operação de salvamento em arquivo, obter-se um tamanho menor do mesmo.

- Nome do nodo: Poderá representar um atributo individual do sistema especialista ou, no caso de tratar-se de um nodo assinalado como objetivo, um valor para um atributo que será um estado meta dentro do espaço de busca.
- Tipo do nodo: Poderá ser:
    - Univalorado: quando só permite uma instanciação, ou seja, um nodo poderá assumir um único valor por vez quando de um consulta.
    - Multivalorado: Mais de uma instanciação é permitida, ou seja, um nodo poderá assumir mais de um valor para uma dada consulta.
    - Numérico: O nodo só permite que os arcos assumam valores numéricos.
- Objetivo: Poderá assumir valor 'Sim' ou 'Não'. Quando o nodo é um objetivo, ele é um estado meta a ser alcançado pelo sistema especialista (Na verdade um valor para um atributo que representa o estado meta).
- Pergunta associada ao nodo: Permitido somente para nodos que não são 'objetivo' – As respostas a estas perguntas, na implementação do sistema especialista, é que permitirão encaminhamento dentro do espaço de busca para que um estado meta seja alcançado (decisão de qual caminho (aresta) será escolhida para permitir a navegação no fluxograma). No processo de consulta a um sistema especialista, uma variável, quando colocada em evidência no quadro negro para refutação de uma hipótese, aparece para o usuário através de uma pergunta e não por seu nome.
- Motivo da pergunta: Seu preenchimento é opcional. Quando preenchido justificará a pergunta formulada para referenciar aquele nó. Quando o campo fica em branco, no formulário de propriedades dos nodos, o sistema especialista, quando implementado, geralmente apresenta justificação default para o uso daquele nodo como componente de uma dada regra para atingir uma determinada meta.

Cada nó, que não seja definido como 'objetivo', corresponde quando da implementação do sistema especialista a atributos. Atributos são variáveis que poderão assumir uma ou múltipla instanciação no decorrer de uma consulta – univaloradas ou multivaloradas – podendo também assumir valor numérico.

As informações para inserção de nodos precisaram de validação o que é obtido com o clique em um botão de inclusão de dados, para com isto, poderem ser gravados na tela de desenho do TPaintBox.

Após validação dos nodos são desenhados os ramos ligando um nó a outro (valores assumidos pelos atributos). O nó, a partir do qual parte a seta, é denominado pai e os nodos

destinos serão os filhos. O sistema permite caminho duplo, ou seja, um nodo pode ao mesmo tempo ser pai e filho de outro nodo.

Os ramos ou arestas apresentam propriedades, quais sejam:

- Valor: Cada ramo representa um valor relacionado a um atributo e o conjunto de ramos corresponde à lista de valores assumidos pelo nodo pai.
- Operador: São operadores relacionais: =,>,<>,>= dentre outros.
- É permitido, na eventualidade do nodo origem tratar-se de um atributo do tipo numérico, que os valores armazenados nos arcos possam apresentar ao invés de um único número, uma faixa de intervalo numérico.
- Não é permitido que um nodo que tenha a propriedade 'objetivo' marcada como verdadeira seja pai de outro nó qualquer. Os nodos com propriedade definida como 'objetivo' serão sempre folhas.

Os nodos são os elementos constituintes das regras (as regras são formados por encadeamento de nodos), e poderão representar quando da implementação do sistema especialista:

- Variáveis: São obtidas a partir dos nodos marcados como não objetivos. São constituintes das premissas das regras. Não serão exibidas pelo sistema especialista, no módulo de consulta, a partir dos seus nomes, e sim, a partir das perguntas permitidas para cada nó (motor de inferência), no entanto, na confecção das regras serão utilizados os seus nomes e os valores assumidos serão tantos quantos forem os números de arcos gerados do nodo.
- Objetivos: Não serão variáveis independentes. Na verdade, serão valores de uma variável 'objetivo' (Por default esta variável recebe o nome de 'resultado', porém, poderá assumir qualquer outro nome de acordo com a vontade do implementador do sistema) e serão mostrados como cabeças das regras. Serão exibidas pelo sistema especialista como conclusão - Regras de conclusão.

Os nodos assinalados como 'objetivo' serão, portanto, dentro do SE, os valores instanciados por uma variável por default denominada 'resultado', aparecerá na cabeça das regras e para definição dos valores assumidos por este atributo será empregado somente o sinal "=" que assumirá valor de atribuição e não o de igualdade.

De acordo com o tipo de variável pode-se, quando da implementação do sistema especialista, ter-se a obtenção de uma única resposta (variável tipo objetivo univalorada) ou mais de uma resposta (variável tipo objetivo multivalorada).

O fator de confiança, representado em termos percentuais (0% a 100%), indica a probabilidade de um ou mais fatos simultâneos ocorrerem para que uma determinada conclusão seja obtida. Este valor não é gerado pelo 'Sistema Gerador de Regras' (SGR), no entanto se este fator for disponível a partir de outras fontes poderá ser acrescentado manualmente às regras geradas quando da implementação do sistema especialista.

Para exemplificação e melhor entendimento das estruturas de dados, bem como para elaboração do sistema especialista (Veja o Capitulo 5) confeccionado a partir do relatório gerado pelo SGR, será empregado o fluxograma da Figura 3. 1 elaborada para escolha de melhor dieta para o paciente de acordo com a sua condição orgânica (à proporção que se for avançando será montado o relatório que gerará as variáveis e as regras que permitirão confeccionar um pequeno sistema especialista que possibilitará ao aluno aprender mais facilmente, de forma interativa, a escolher a dieta mais adequada para um paciente cirúrgico):

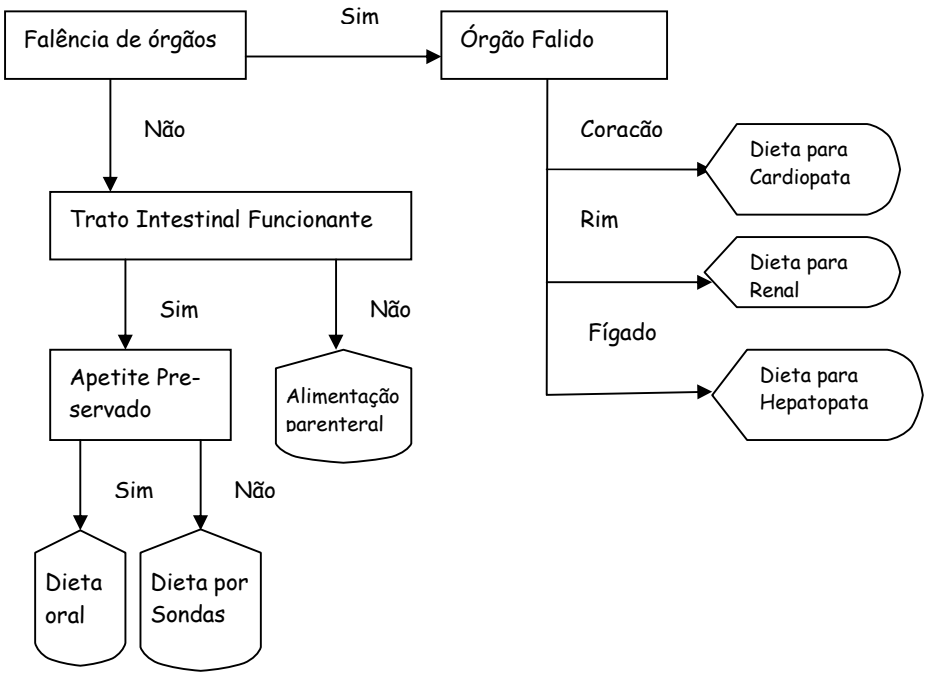


Figura 3. 1 – Fluxograma para escolha da melhor dieta para um paciente cirúrgico

3.3.1 Detalhamento do sistema

Para que seja possível a geração de fatos e regras, a partir do desenho de um fluxograma de decisão, foi desenvolvido um programa que permite a utilização da estrutura de

dados grafo num problema real. O desenho será armazenado num grafo e será possível executar diversas operações sobre esse grafo, como: inclusão, alteração e exclusão de nodos, atribuição e exclusão de valores para os arcos que conectarão os nodos, bem como gravar e restaurar essa estrutura em um arquivo. O resultado do sistema será um relatório que permitirá a alimentação de um "shell" de sistema especialista baseado em regras que possibilitará ao aluno de bases da técnica cirúrgica, de modo interativo, respondendo as perguntas quando do modo de consulta do sistema especialista simulação de casos reais representados por estes fluxogramas.

Desenvolver-se-á um sistema completo e, com o relatório gerado pelo software, será construído, a partir de qualquer "shell", um pequeno módulo de sistema especialista.

Por ter sido elaborado adotando o idioma português como base, para tornar a tarefa de construção de um sistema especialista fácil, mesmo para quem não tem conhecimentos amplos em informática, por ser adequadamente documentado e ser amplamente empregado nas Universidades brasileiras no ensino de Inteligência Artificial fez-se opção pelo "shell" Expert Sinta desenvolvido pelo LIA – Laboratório de Inteligência Artificial da UFC - Universidade Federal do Ceará, para implementação do sistema especialista (Veja o capitulo 5).

3.4 Especificação do problema

O problema em questão é a criação de um programa que utiliza uma estrutura para armazenar alguns dos principais dados de um fluxograma de decisão de uma conduta médica, que possibilita a criação de um gerador de regras para alimentar um sistema especialista que irá proporcionar ao aluno da disciplina bases da técnica cirúrgica, interagindo com o sistema, sedimentar o assunto que foi ensinado em sala de aula ou contido em impressos médicos. O programa desenvolvido para trabalhar esse problema é um gerador de regras para confecção de sistemas especialistas.

Esse programa é responsável pela criação da estrutura de um grafo valorado baseado em um fluxograma de decisão e pela execução de diversas operações sobre o mesmo. A criação da estrutura, ou do grafo propriamente dito, consiste em operações de inclusão, alteração e exclusão de nodos, e inclusão, alteração e exclusão de ligações entre os nodos (valores apresentados pelos nodos antecessores) presentes no desenho que representa o fluxograma de decisão. É possível também que se altere a posição relativa na tela do com-

putador de cada nodo na caixa de desenho arrastando um nodo de um ponto para outro da tela.

As operações executadas irão resultar na saída de variáveis de SE com todas as suas propriedades e regras que são obtidas a partir do encadeamento entre os nodos. Os dados fornecidos são os seguintes:

- Todas as variáveis e suas propriedades: nome, valores, tipo, perguntas formuladas e motivo da mesma (opcional).
- Valores e sinais de atribuições que permitem a ligação de um nodo a outro. Funciona em IA como espaço de busca.

A estrutura de dados escolhida para armazenar todas os nodos e todos os seus valores, que possibilitam a ligação com um outro nodo, foi o grafo valorado, implementado através de duas listas encadeadas dinamicamente.

Cada nodo representa uma variável e suas propriedades, no entanto, quando a propriedade 'objetivo' é definida como 'verdadeiro' os nodos marcados como tal representam valores de uma variável 'objetivo' que por "default" tem o nome de 'resultado' (podendo assumir qualquer outro nome de acordo com a vontade do usuário).

3.5 Descrição das estruturas de dados utilizadas

A estrutura de dados que armazena o mapa contendo o fluxograma de decisão consiste num grafo valorado implementado utilizando duas listas encadeadas dinamicamente: a primeira com encadeamento simples e a segunda com encadeamento duplo. A primeira contém um campo que é ponteiro para um elemento da segunda lista. Observe as descrições abaixo:

3.5.1 Lista principal - Simplesmente encadeada

Nome do campo	***Tipo***	***Descrição***
CodNodo	*Caractere [2]*	*Armazena o código do nodo da primeira lista. É utilizado para simplificar o desenho do nodo no mapa, pois apenas o código do nodo é exibido, e também para possibilitar a geração de um arquivo de saída menor, uma vez que todos os dados de um nodo são gravados num único registro e se posteriormente for necessário referência a ele no arquivo é utilizado somente o seu código. A estrutura do arquivo de saída será mais bem detalhada adiante.*
NomeNodo	*Caractere [100]*	*Armazena o nome do nodo.*

Objetivo	*Lógico*	*Quando 'verdadeiro' faz com que o nodo represente valores de uma variável que é objetivo do sistema especialista Quando 'falso' cada nodo representará uma variável independente com todas as suas propriedades.*
Tipo	*Enumerado*	*Definição do tipo de variável quando o sistema especialista for implantado poderá assumir os valores:* - *Numérico* - *Univalorada: Dentro do SE só permitirá uma instanciação por vez para a variável.* - *Multivalorada: mais de um valor pode ser instanciado por vez.*
Pergunta	*Caractere*	*Permite que quando da instanciação da variável no sistema especialista uma pergunta seja formulada. Somente nodos não objetivos permitem perguntas.*
Motivo	*Caractere*	*Preenchimento opcional. Explica o motivo para formulação de uma dada pergunta. Quando não preenchido normalmente o "shell" de sistema especialista tem o default (mecanismo de explicação do "shell" de SE).*
Relacionamento1	*Caractere [2]*	*É utilizada somente na geração do arquivo de saída. Ver descrição do arquivo de saída.*
Valor1	*Caractere*	*É utilizada somente na geração do arquivo de saída. Ver descrição do arquivo de saída.*
Relacionamento2	*Caractere [2]*	*É utilizada somente na geração do arquivo de saída. Ver descrição do arquivo de saída.*
Valor2	*Caractere*	*É utilizada somente na geração do arquivo de saída. Ver descrição do arquivo de saída.*
xinodo, yinodo	*Inteiro*	*Armazenam a posição inicial de cada um dos nodos no mapa.*
xfnodo, yfnodo	*Inteiro*	*Armazenam a posição final de cada um dos nodos no mapa. A posição final é igual a inicial mais um valor constante que define o diâmetro da circunferência que será desenhada no mapa.* *Esses quatro campos são de extrema importância, pois é através deles que o mapa é exibido graficamente na tela.*
Prox	*Ponteiro para a Lista Principal*	*Aponta para o próximo nó cadastrado.*
PrimAdj	*Ponteiro para a Lista de Adjacentes*	*Aponta para um elemento da lista de adjacentes que é o primeiro nó adjacente ao da lista principal.*
UltAdj	*Ponteiro para a Lista de Adjacentes*	*Aponta para o último nó adjacente a da lista principal.*

Tabela 3. 1– Lista principal

3.5.2 Lista de adjacentes – Lista duplamente encadeada

Utilizada para armazenar os ponteiros que apontam para cada nodo adjacente ao nodo da lista principal e o respectivo valor atribuído ao nodo origem que permite gerar o nodo destino.

Contém na sua estrutura os seguintes campos

Nome do campo	*Tipo*	*Descrição*
Nodo	*Ponteiro para a lista principal*	*Aponta para um elemento da lista principal (nodo). Isto é, aponta para um endereço onde está localizada toda a informação de um determinado nodo. Esse nó apontado é adja-*

		cente ao nó que está localizada na lista principal e que tem o seu campo PrimAdj apontando para o primeiro elemento dessa lista.
Relacionamento1	*Caractere [2]*	*Sinal atribuído a um determinado valor da variável.*
Valor1	*Caractere*	*Valores instanciados para as variáveis. Aparecerá nas arestas dos grafos.*
Relacionamento2	*Caractere [2]*	*Sinal atribuído a um determinado valor da variável (valor2) que delimita um intervalo com o valor1. Quando não existe intervalo é uma "string" vazia.*
Valor2	*Caractere*	*Valores instanciados para as variáveis. Aparecerá nas arestas dos grafos. Serve para delimitação de um intervalo numérico feito com o valor1. Quando não existe intervalo é uma "string" vazia.*
Prox	*Ponteiro para a lista de adjacentes*	*Aponta para o próximo elemento dessa lista. Serve para realizar o encadeamento entre os elementos.*
Ant	*Ponteiro para a lista de adjacentes*	*Aponta para o elemento anterior dessa lista Esse campo é necessário, pois a lista é duplamente encadeada.*

Tabela 3. 2 – Lista de adjacentes

O fluxograma da figura 3. 1, que exemplifica a aplicação, é esboçada na tela de desenho como na figura 3. 2 (Grafo valorado) e sua forma estrutural é evidenciada na figura 3. 3.

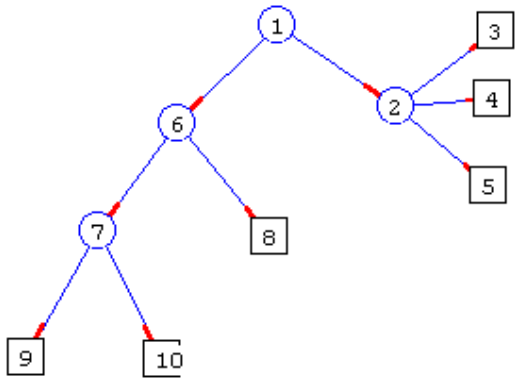


Figura 3. 2 – Forma gráfica desenhado no 'Sistema Gerador de Regras' a partir do fluxograma da Figura 3. 1

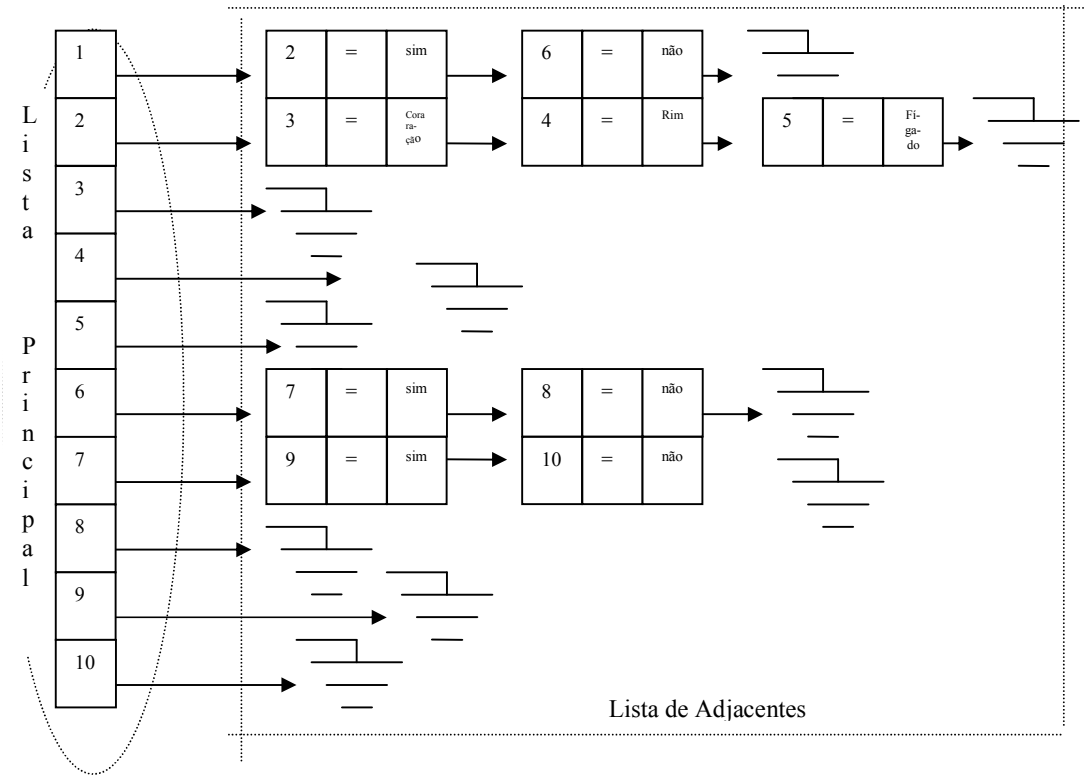


Figura 3. 3 – Forma estrutural do grafo da Figura 3. 2

3.5.3 Estrutura PilhaRegra

É utilizada no momento em que é feito algum caminhamento no grafo. O caminhamento é feito por profundidade, isso quer dizer que cada nodo visitado é colocado nessa pilha, e o seu conteúdo final é o próprio caminho final.

Nodo	*Ponteiro para a lista principal do grafo.*	*Aponta para um determinado nodo do mapa. Esse nó apontado é adjacente ao nó de origem.*
Relacionamento1	*Caractere [2]*	*Sinal atribuído a um determinado valor da variável.*
Valor1	*Caractere*	*Valores instanciados para os nodos do grafo.*
Relacionamento2	*Caractere [2]*	*Sinal atribuído a um determinado valor da variável numérica – só para intervalos.*
Valor2	*Caractere*	*Valores instanciados para os nodos do grafo – só para intervalos numéricos.*
Prox	*Ponteiro para a própria estrutura*	*Utilizado para realizar o*

		encadeamento.

Tabela 3. 3 – Estrutura PilhaRegra

Descrição estruturada do caminhamento por profundidade para formação de regras:

1. *Visita-se o nó N e coloca-o na pilha.*
2. *Para cada nó M adjacente a N*
    a. *Visita-se o nó M*
    b. *Coloca-se M na pilha*
    c. *Atribui N a M (N:=M).*
3. *Se a pilha não estiver vazia, toma-se o nó que está no topo da pilha (nó N) e volta-se para o passo 2 (é uma função recursiva).*

Observe que, como o algoritmo é recursivo, quando o nodo destino é encontrado ele volta para o anterior e continua a pesquisa a partir desse nodo.

Cada caminho encontrado é colocado numa lista encadeada auxiliar denominada lista de regras.

3.5.4 Lista de regras

É uma lista encadeada dinamicamente que contém na sua estrutura os seguintes campos:

Nodo	*Ponteiro para a lista principal do grafo.*	*Aponta para um determinado nó do mapa. Esse nó apontado é adjacente ao nó origem.*
Linha	*Inteiro*	*Contém a linha da Tabela utilizada para exibir todos as regras encontradas.*
Prox	*Ponteiro para a própria estrutura.*	*Utilizado para realizar o encadeamento.*

Tabela 3. 4 – Lista de Regras

A lista de regras funciona como uma tabela. Todos os nodos encontrados durante um caminhamento são colocados nele e cada regra formada é referenciada pelo campo linha. Exemplo:

Vamos supor que para dados três nodos: A, B e C foram encontradas as seguintes regras: "Se A e B então C" e "Se A então C". A regra "Se A e B então C" seria a de linha 1 e a regra "Se A então C" seria a de linha 2.

3.5.5 Lista dos nós origem

É uma lista formada por todos os nodos fontes. Um nó fonte é aquele que não esta em nenhuma das listas de adjacentes de todos os nodos.

Sua estrutura contém os seguintes campos:

NodoOrigem	*Ponteiro para a lista principal do grafo.*	*Aponta para um determinado nó do mapa. Esse nó apontado não faz parte de nenhuma lista de adjacentes.*
Prox	*Ponteiro para a própria estrutura.*	*Utilizado para realizar o encadeamento.*

Tabela 3. 5 – Lista dos nós fontes

3.5.6 Lista dos nós objetivos

É uma lista formada por todos os nodos que são sumidouros. É formada por todos os nodos assinalados como 'objetivo'.

Sua estrutura contém os seguintes campos:

NodoDestino	*Ponteiro para a lista principal do grafo.*	*Aponta para um determinado nó do mapa. Esse nó é marcado com 'Objetivo'.*
Prox	*Ponteiro para a própria estrutura.*	*Utilizado para realizar o encadeamento.*

Tabela 3. 6 – Lista dos nodos objetivos

3.6 Algoritmos principais

3.6.1 Inserção

Insere um novo nodo no grafo. O novo nodo é sempre incluído depois do último cadastrado.

O algoritmo recebe uma estrutura contendo os dados a serem incluídos e também aufere um ponteiro para o primeiro e um ponteiro para o último elemento do grafo (esses dois ponteiros são passados por referência). Abaixo está o algoritmo já implementado.

```
{Executado quando o Botão Inclui é clicado}
procedure TFrmMapa.BtnIncluiClick(Sender: TObject);
var Opcao_tipo:Tipovariavel;
    objetivo:boolean;
begin
  if (EdCodNodo.Text='') or (EdCodNodo.Text='-') then begin
     {Não deixa que o código do Nodo receba o caractere '-' pois ele é
usado na formatação do arquivo de saída}
     ShowMessage('O Campo código está vazio ou preenchido incorretamen-
te.');
     EdCodNodo.SetFocus ;
     exit
     end
  else if InGrafo(PrimGrafo,EdCodNodo.Text) then begin
     {Se já existir esse código}
     ShowMessage('Esse código já existe.');
     exit;
  end;
    {Recebe o valor para Tipo de variável - Numérica, Univalora-
da,Multivalora}
  If RgTipo.ItemIndex = 1 then
     Opcao_tipo := numerica
     else
         if RgTipo.ItemIndex = 0 then
            Opcao_tipo:= Univalorada
                       else
                           if RgTipo.ItemIndex = 2 then
                              Opcao_tipo:= multivalorada;
  {Recebe o valor para Objetivo}
   if RgObjetivo.ItemIndex = 0 then
      objetivo := true
                else objetivo :=false;
   {Não deixa que o nodo fique sem um nome}
   while EdNomeNodo.Text='' do begin
       ShowMessage('O Campo Nome da Variável deve estar preenchido');
       EdNomeNodo.SetFocus;
       exit;
   end;
    if RGObjetivo.ItemIndex = 1 then begin
       while EdPergunta.Text='' do begin
           ShowMessage('É necessária uma pergunta para que a Variável se-
ja instanciada no SE');
           EdPergunta.SetFocus;
           exit;
      end;
   end;
  {Atualiza a variável RegGrafo com os valores do formulário}
  RegGrafo.CodNodo:=EdCodNodo.Text;
  RegGrafo.nome_nodo:=EdNomeNodo.Text;
  RegGrafo.Opcao_tipo := opcao_tipo;
  RegGrafo.Objetivo := objetivo;
  RegGrafo.Pergunta := EdPergunta.Text;
  RegGrafo.Motivo := EdMotivo.Text;
```

```
  RegGrafo.relacionamento1 := '';
  RegGrafo.nome_valor1:='';
  RegGrafo.relacionamento2 := '';
  RegGrafo.nome_valor2:='';
  RegGrafo.xiNodo:=10;
  RegGrafo.yiNodo:=10;
  RegGrafo.xfNodo:=10+TAMNODO;
  RegGrafo.yfNodo:=10+TAMNODO;
  RegGrafo.Prox:=nil;
  RegGrafo.PrimAdj:=nil;
  RegGrafo.UltAdj:=nil;
  {Inclusão no Grafo}
  IncluiGrafo(PrimGrafo,UltGrafo,RegGrafo);
  AtualizaCmbGrafo;
  {Desenho da Variável no Mapa}
  DesMapa;
  LimpaFormNodo;
  EdCodNodo.SetFocus;
end;
```

3.6.2 Remoção

Remove um elemento do grafo. Esse algoritmo remove o elemento que está selecionado no momento, ou seja, o elemento apontado pelo ponteiro PAtualGrafo. Além de remover esse elemento do grafo, verifica a existência dele em todas as listas de adjacências, e se estiver presente em alguma delas, é imediatamente excluído. Veja o algoritmo já implementado:

```
{Executado quando o botão Exclui é clicado}
{Exclui do grafo o nodo que está sendo apontado por PAtualGrafo }
 procedure TFrmMapa.BtnExcluiClick(Sender: TObject);
var PAntGrafo,PAux:PTRGrafo;
begin
  if PrimGrafo=nil then begin
     ShowMessage('Não há Nenhum Nodo Cadastrado');
     exit;
  end;
  PAux:=PrimGrafo;
  PAntGrafo:=PrimGrafo;
  {Primeiro Exclui todas as ligações dele}
  ExcluiLigacoes(PAtualGrafo);
  if PAtualGrafo=PrimGrafo then begin
     {Se for o primeiro do grafo}
     PrimGrafo:=PAtualGrafo^.Prox;
     PAtualGrafo^.Prox:=nil;
     Dispose(PAtualGrafo);
     PAntGrafo:=nil;
     end
  else while(PAux<>nil) do begin
       if PAtualGrafo=PAux then begin
          PAntGrafo^.Prox:=PAux^.Prox;{Próximo do anterior recebe próximo
do PAux}
          PAux^.Prox:=nil; {Próximo do PAux recebe nil}
          if PAux=UltGrafo then UltGrafo:=PAntGrafo; {Se o excluído for o
```

```
                                                    último, atualiza o último}
          Dispose(PAux);
          break;
          end
       else begin {Continua varrendo o grafo}
          PAntGrafo:=PAux;
          PAux:=PAux^.Prox;
        end;
  end;
  if PAntGrafo<>nil then PAtualGrafo:=PAntGrafo {Atual aponta para o an-
terior do excluído}
  else PAtualGrafo:=PrimGrafo; {Para não dar pau se o excluído for o pri-
meiro do grafo}
  AtualizaCmbGrafo;
  DesMapa;
  LimpaListaOrigem(PrimListaOrigem,UltListaOrigem);
  LimpaListaDestino(PrimListaDestino,UltListaDestino);
  LimpaListaRegra(PrimListaRegra,UltListaRegra); {Para não dar pau se a
                    variável excluída fazia parte de uma das regras}
  AtuJanNodo;
end;
```

3.6.3 – Algoritmo para alteração de um valor do formulário

Este algoritmo é utilizado para alterar propriedades de um nodo selecionado no grafo. Abaixo está o algoritmo já implementado.

```
{Executado quando o botão Altera é Clicado}
procedure TFrmMapa.BtnAlteraClick(Sender: TObject);
var Opcao_tipo:Tipovariavel;
    objetivo:boolean;
begin
  if PrimGrafo=nil then begin
     ShowMessage('Não Há Nenhuma Variável Cadastrada');
     exit;
     end
  else if EdCodNodo.Text='' then begin
       ShowMessage('O Campo código da Variável deve estar preenchido');
       exit;
  end;
   If RgTipo.ItemIndex = 1 then
     Opcao_tipo := numerica
     else
         if RgTipo.ItemIndex = 0 then
            Opcao_tipo:= Univalorada
                       else
                           if RgTipo.ItemIndex = 2 then
                              Opcao_tipo:= multivalorada;
  {Recebe o valor para Objetivo}
   if RgObjetivo.ItemIndex = 0 then
      objetivo := true
               else objetivo :=false;
    {Atualiza os dados de PAtualGrafo}
  PAtualGrafo^.CodNodo:=EdCodNodo.Text;
  PAtualGrafo^.nome_nodo:=EdNomeNodo.Text;
  PAtualGrafo^.Pergunta :=EdPergunta.Text;
  PAtualGrafo^.Motivo := EdMotivo.Text ;
  PAtualGrafo^.Objetivo:= objetivo;
```

```
  PAtualGrafo^.Opcao_tipo := opcao_tipo;
  AtualizaCmbGrafo;
end;
```

3.6.4 Algoritmo para limpar o formulário de dados

Limpa o formulário para permitir a entrada de novos dados. Abaixo está o algoritmo já implementado.

```
{Executado quando o botão <Limpa> é clicado}
procedure TFrmMapa.BtnLimpaClick(Sender: TObject);
begin
  {Limpa os objetos}
  CmbGrafo.Text:='';
  EdCodNodo.Text:='';
  EdNomeNodo.Text:='';
  EdPergunta.Text :='';
  EdMotivo.Text :='';
  RGTipo.ItemIndex:=0;
  RGObjetivo.ItemIndex:=1;
  EdCodNodo.Setfocus;
end;
```

3.6.5 Algoritmo de atribuição de valores entre os nodos

Algoritmo que permite a atribuição ou alteração do valor que liga um nodo a outro. Abaixo está o algoritmo já implementado.

```
{Exucutado quando o Botão Atribui é clicado
 Insere/Altera um valor entre dois nodos adjacentes}
procedure TFrmMapa.BtnAplDistClick(Sender: TObject);
var POrigem,PDestino:PTRGrafo;
begin
  while (CmbOrigem.Text='') or (CmbDestino.Text='') do begin
     ShowMessage('É necessário o preenchimento dos campos Origem e Desti-
no');
     exit;
  end;
  while (CmbOrigem.Text=CmbDestino.Text) do begin {Não permite o laço}
     ShowMessage('A variável origem é igual a variável destino');
     exit;
  end;
nome_valor1:=edValor1.Text;
relacionamento1 := CmbRelacionamento1.Text ;
nome_valor2:=edValor2.Text;
relacionamento2 := CmbRelacionamento2.Text ;
  {Pesquisa no grafo os nodo de origem e de destino, atribuindo os
   respectivos ponteiros}
  POrigem:=AtribPtrGrafo(PrimGrafo,CmbOrigem.Text); {Retorna um ponteiro
                                                     para POrigem}
  PDestino:=AtribPtrGrafo(PrimGrafo,CmbDestino.Text); {Retorna um pontei-
ro
                                                       para PDestino}
   AtribDistGra-
fo(POrigem,PDestino,relacionamento1,nome_valor1,relacionamento2,nome_valo
r2); {Função de inclusão na lista de adjacência}
  if RadioSim.Checked=True then {Se for "mão-dupla" (não-dirigido) execu-
```

```
ta a mesma função trocando a origem pelo destino e o destino pela origem}
     AtribDisGrafo(PDestino,POrigem,relacionamento1,nome_valor1,
relacionamento2,nome_valor2);
     DesMapa;
  { O preenchimento do campo valor1 é obrigatório}
  while EdValor1.Text = '' do begin
     ShowMessage('É obrigatório o preenchimento do valor');
     edValor1.SetFocus ;
     exit;
  end;
   {Se for permitido Intervalo é obrigatório o preenchimento da caixa de
texto2 com um valor numérico}
    if (RgIntervalo.ItemIndex = 0) then begin
       while (EdValor2.Text = '') do begin
          ShowMessage('É obrigatório o preenchimento do valor');
          edValor2.SetFocus ;
          exit;
       end;
  end;
 DesMapa;
 end;
```

3.6.6 Algoritmo para exclusão de arcos

Permite que seja feita a exclusão do arco que liga um nodo ao outro. Abaixo está o algoritmo já implementado.

```
{Executado quando o botão de exclusão de caminho é clicado
 Exclui uma ligação entre dois nodos}
procedure TFrmMapa.BtnExcClick(Sender: TObject);
var POrigem,PDestino:PTRGrafo;
    PAuxAdj:PTRListaAdj;
begin
  {Atribui os Ponteiros para o nodo origem e destino}
  POrigem:=AtribPTRGrafo(PrimGrafo,CmbOrigem.Items[CmbOrigem.ItemIndex]);
  PDestino:=AtribPTRGrafo(PrimGrafo,
      CmbDestino.Items[CmbDestino.ItemIndex]);
  if (POrigem=nil) or (PDestino=nil) then exit;
  PAuxAdj:=POrigem^.PrimAdj;
  while(PAuxAdj<>nil) do begin {começa a pesquisa na lista de adjacência}
       if PAuxAdj^.Nodo=PDestino then begin {Se encontrar o PDestino na
lista de adjacência do POrigem}
          ExcluiCaminho(POrigem^.PrimAdj,POrigem^.UltAdj,PAuxAdj);
          break;
          end
       else PAuxAdj:=PAuxAdj^.Prox;
  end;
  LimpaListaOrigem(PrimListaOrigem,UltListaOrigem);
  LimpaListaDestino(PrimListaDestino,UltListaDestino);
  LimpaListaRegra(PrimListaRegra,UltListaRegra); {Para não dar pau se ex-
cluir uma ligação que está presente na lista}
  DesMapa;
end;
```

3.6.7 Algoritmo para formação da lista de origem

Para formação da lista de origem, ou seja, a lista contendo todos os nodos fontes da

estrutura, o algoritmo já implementado empregado é:

```
{Procedimento para formar a lista de Origem}
Procedure TFrmMapa.FormarListaOrigem;
var PAux:PTRGrafo;
Begin
PAux:=PrimGrafo; {Para iniciar no Começo do Grafo};
while PAux<> nil do begin
  if not EstaNosAdjacentes(PAux) then
  IncListaOrigem(PrimListaOrigem,UltListaOrigem,PAux);
  PAux:=Paux.Prox;
 end;
end;
```

Perceba que é imprescindível a verificação se o nodo faz parte da lista de adjacentes dos outros nodos o que só é possível a partir de:

```
{Utilizada para saber se um nó esta na lista de todos os adjacentes}
Function TfrmMapa.EstaNosAdjacentes(var Nodo:PTRGrafo):boolean;
var PAuxGrafo:PTRGrafo;
    PAuxAdj:PTRListaAdj;
    Encontrado:boolean;
begin
  Encontrado:=false;
  PAuxGrafo:=PrimGrafo; {Para iniciar no Começo do Grafo};
  while(PAuxGrafo<>nil) and (not encontrado) do begin {começa a pesquisa
no grafo}
       PAuxAdj:=PAuxGrafo^.PrimAdj;
       while(PAuxAdj<>nil) do begin {começa a pesquisa na lista de adja-
centes}
            if PAuxAdj^.Nodo=PAuxGrafo then begin
               Encontrado:=true;
               break;
               end
            else PAuxAdj:=PAuxAdj^.Prox;
       end;
  PAuxGrafo:=PAuxGrafo^.Prox;
  end;
  EstaNosAdjacentes:=Encontrado;
end;
```

3.6.8 Algoritmo para formação da lista de destino

Este algoritmo serve para formação da lista dos nodos que são sumidouros do grafo, ou seja, os objetivos do sistema. Abaixo está o algoritmo já implementado.

```
Procedure TFrmMapa.FormarListaDestino;
var PAux:PTRGrafo;
Begin
PAux:=PrimGrafo; {Para iniciar no Começo do Grafo};
while PAux<> nil do begin
 if (PAux.Objetivo=true) then
  IncListaDestino(PrimListaDestino,UltListaDestino,PAux);
  PAux:=PAux.Prox;
 end;
end;
```

3.6.9 Algoritmo para formação da lista de regra

Quando se quer obter o relatório para confecção do sistema especialista, parte-se do algoritmo (já implementado):

```
{Executado quando o botão de resultados é clicado}
procedure TFrmMapa.BtnResultadoClick(Sender: TObject);
begin
DefinicaoObjetivo;
FrmResultado.RERelatorio.Lines.Clear;
Numero:=1;{Atribui a 1ª regra o número 1}
  {Obtém todas as variáveis do sistema especialista}
  VariaveisDoSistema;
  {Obtém todas as Regras do sistema especialistas baseadas no Grafo}
  FrmMapa.FormaRegra ;
  {Exibe os Fatos e regras que alimentarão o sistema especialista}
  FrmResultado.ShowModal;
  DesMapa;
end;
```

Para obtenção dos fatos que constituem o sistema especialista emprega-se:

```
{Procedimento para relacionar as variáveis e seus respectivos valores que
irão formar o sistema especialista}
procedure TFrmMapa.VariaveisDoSistema;
var PAuxGrafo:PTRGrafo;
    PAuxAdj:PTRListaAdj;
    Opcao:string;
begin
PAuxGrafo:=PrimGrafo;
{Variáveis que não são objetivos do Sistema}
FrmResultado.RERelatorio.Lines.Add('VARIÁVEIS');
FrmResultado.RERelatorio.Lines.Add('');
while PAuxGrafo <> nil do begin
  if PAuxgrafo.Objetivo = false then begin
       {Para receber o valor do tipo de variável - transformação de um
tipo enumerado em uma string}
        If PAuxGrafo.Opcao_tipo = numerica then
           opcao:='numérica'
           else if PAuxGrafo.Opcao_tipo = univalorada
            then opcao:= 'univalorada'
              else opcao:= 'multivalorada';
       FrmResultado.RERelatorio.Lines.Add('     '+'NOME:
'+PAuxGrafo.Nome_nodo);
       FrmResultado.RERelatorio.Lines.Add('        '+'OBJETIVO: NÃO');
       FrmResultado.RERelatorio.Lines.Add('        '+'TIPO: '+ opcao);
       FrmResultado.RERelatorio.Lines.Add('        '+'PERGUNTA:
'+PAuxGrafo.Pergunta);
       FrmResultado.RERelatorio.Lines.Add('        '+'MOTIVO DA PERGUNTA:
'+
                                                  PAuxGrafo.Motivo);
       FrmResultado.RERelatorio.Lines.Add('        '+ 'VALORES');
       PAuxAdj:=PAuxGrafo^.PrimAdj;
        {Para encontrar os Valores de Cada Variável Visitamos os nodos
adjacentes}
            while(PAuxAdj<>nil) do begin {começa a pesquisa na lista de
adjacentes}
            FrmResultado.RERelatorio.Lines.Add('
'+PAuxAdj.Relacionamento1+
                ' '+PAuxAdj.nome_valor1+'
'+PAuxAdj.Relacionamento2+' '+
```

```
                PAuxAdj.nome_valor2);
              PAuxAdj := PAuxAdj^.Prox;
               end;
    FrmResultado.RERelatorio.Lines.Add('');
  end;
  PAuxGrafo:=PAuxGrafo.Prox;
end;
  {Variáveis que são objetivos do Sistema}
  FrmResultado.RERelatorio.Lines.Add('');
  FrmResultado.RERelatorio.Lines.Add('     '+'NOME: '+ Resultado);
  FrmResultado.RERelatorio.Lines.Add('        '+'OBJETIVO: SIM');
  FrmResultado.RERelatorio.Lines.Add('        '+'Tipo: '+ TipoObjetivo);
  FrmResultado.RERelatorio.Lines.Add('        '+ 'VALORES');
  PAuxGrafo:=PrimGrafo;
  {Os Valores são os nodos marcados com Objetivo}
  While PAuxGrafo<> nil do begin
        if PAuxGrafo.Objetivo = true then
              FrmResultado.RERelatorio.Lines.Add('            '+'=
'+PAuxGrafo.Nome_nodo);
        PAuxGrafo:=PAuxGrafo.Prox;
  end;
end;
```

As regras são formadas a partir de:

```
{Finalmente o Caminhamento entre dois nodos - Procedimento para formação
das Regras}
procedure TFrmMapa.FormaRegra;
var PTROrigem,PTRDestino:PTRGrafo;
    PAuxListaOrigem:PTRListaOrigem;
    PAuxListaDestino:PTRListaDestino;
begin
   {Limpa Lista de regras, lista de origem e de destino};
  LimpaListaRegra(PrimListaRegra,UltListaRegra);
  LimpaListaOrigem(PrimListaOrigem,UltListaOrigem);
  LimpaListaDestino(PrimListaDestino,UltListadestino);;
  FrmResultado.RERelatorio.Lines.Add('');
  FrmResultado.RERelatorio.Lines.Add('');
  FrmResultado.RERelatorio.Lines.Add('REGRAS');
  FrmResultado.RERelatorio.Lines.Add('');
  FrmResultado.RERelatorio.Lines.Add('');
  {Atualiza ponteiros para os nodos de origem e destino}
  FormarListaOrigem;
  FormarListaDestino;
  if PrimListaDestino = nil then begin
     showmessage('O Grafo não apresenta objetivos definidos');
     exit;
  end;
  PAuxListaOrigem:=PrimListaOrigem;
    while PAuxListaOrigem  <> nil do begin
      PTROrigem:=PAuxListaOrigem^.NodoOrigem;
      PAuxListaDestino:=PrimListaDestino;
      while PAuxListaDestino <> nil do begin
          PTRDestino:=PAuxListaDestino^.NodoDestino;
          if (PTROrigem=nil) or (PTRDestino=nil) then  exit;
          ConstruirRegras(PTROrigem,PTRDestino); {Executa o procedimento
recursivo forma regra}
      PAuxListaDestino:=PAuxListaDestino.Prox;
      end;
  PAuxListaOrigem:=PAuxListaOrigem.Prox;
  end;
```

```
  end;
```

O algoritmo usado para percorrer o grafo foi um algoritmo de profundidade, isto é, um algoritmo que usa uma pilha como estrutura auxiliar. Cada nodo do caminho é empilhado na pilha. O conteúdo final da pilha é a própria regra. É um procedimento recursivo. Observe a sua implementação:

```
   {Procedimento que realiza o caminhamento entre dois nodos do grafo.
 Recebe o nodo origem, o nodo destino e a estrutura que receberá os re-
sultados da formação de regras.
 Caminhamento por PROFUNDIDADE - Estrutura auxiliar: Pilha}
procedure TFrmMapa.ConstruirRegras(POrigem,PDestino:PTRGrafo);
var PAuxAdj:PTRListaAdj;
    begin
  if POrigem=PDestino then begin {Se chegou ao nodo}
     Empilha(POrigem); {Empilha a origem}
     AtuJanRegras; {Atualiza janela de Regras}
     Desempilha;
     end
  else begin
     Empilha(POrigem); {Empilha a origem}
     PAuxAdj:=POrigem^.PrimAdj; {começa a pesquisar os nodos adjacentes}
     while PAuxAdj<>nil do begin
          if not Visitada(PAuxAdj^.Nodo) then begin {se não estiver na
pilha}
                  Topo^.relacionamento1:=PAuxAdj^.relacionamento1; {A-
tribui o sinal ao valor do nodo}
                 Topo^.nome_valor1 :=PAuxAdj^.nome_valor1; {Atribui o
valor do nodo}
                  Topo^.relacionamento2:=PAuxAdj^.relacionamento2; {A-
tribui o sinal ao valor do nodo - Intervalo}
                 Topo^.nome_valor2 :=PAuxAdj^.nome_valor2; {Atribui o
valor do nodo - Intervalo}
                 ConstruirRegras(PAuxAdj^.Nodo,PDestino); {Executa for-
mação de regras com a nova Origem **Recursão}
         end;
           PAuxAdj:=PAuxAdj^.Prox; {Vai para o próximo da lista de adja-
centes}
     end;
     Desempilha;
  end;
end;
```

3.7 Fluxo de dados durante um encaminhamento (Figura 3. 4)

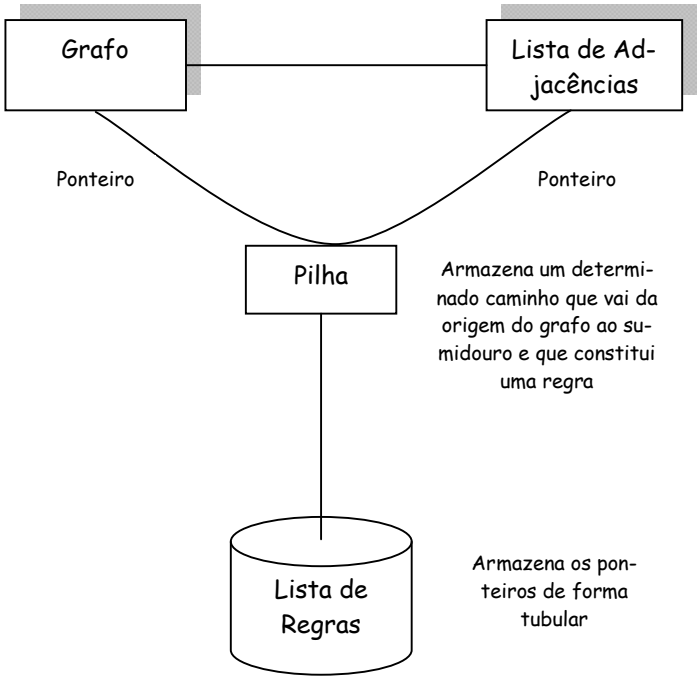


Figura 3. 4 – Fluxograma de dados durante um encaminhamento

3.8 Descrição do arquivo gerado

O programa permite que a estrutura formada seja gravada em um arquivo. Esse arquivo é um arquivo binário (tipado) e o tipo (estrutura) do seu registro é igual ao tipo grafo (Observe o código fonte – Anexo II). Os registros foram armazenados da seguinte maneira:

Primeiramente, todos os nodos armazenados no grafo, com todas as suas informações (exceto a lista de adjacentes) são gravados nos primeiros registros. Após a gravação de todas eles, é incluído um registro que armazena no campo CodNodo o caractere '-' (hífen). Far-se-á referência a esse registro como delimitador. Esse delimitador indica que acabou a gravação dos dados dos nodos. Após o registro delimitador, é incluídos um nodo origem, e os próximos registros incluídos serão as adjacentes a esse nodo origem. Quando os nodos adjacentes a essa origem acabarem, será incluído novamente um delimitador. Esse procedimento é repetido até que todas os nodos de origem e seus respectivos adjacentes tenham sido gravadas. O arquivo termina sempre com um delimitador.

É importante notar que os registros que armazenam os nodos adjacentes contêm preenchidos apenas cinco campos: o código do nodo (CodNodo), o(s) sinal(is) de relacionamento e o valor(es) de cada aresta. É o único momento em que os campos: Relacionamento1, Valor1, Relacionamento2 e Valor 2 são utilizados. O armazenamento do arquivo dessa forma resultou num arquivo de tamanho pequeno, uma vez que os nodos com todos os seus dados são gravadas apenas uma vez e depois a referência a elas é feita apenas pelo seu código.

Observe o desenho de todos os registros utilizados para armazenar o fluxograma exemplificado anteriormente:

Registro 1	**Registro 2**	**Registro 3**
CodNodo: 1 *NomeNodo: Falência de Órgãos* *Objetivo: Não* *Tipo: Univalorada* *Pergunta: O Paciente tem falência de algum órgão?* *Motivo:* *Relacionamento1:* *Valor1:* *Relacionamento2:* *Valor2:* *yicid,xicid:* <val. Inteiros> *yfcid,xfcid:* <val. Inteiro> *Prox:* nil *PrimAdj:* nil *UltAdj:* nil	*CodNodo:* 2 *NomeNodo: Condição Especial* *Objetivo: Não* *Tipo:* Univalorada *Pergunta: Qual o órgão Falido?* *Motivo:* *Relacionamento1:* *Valor1:* *Relacionamento2:* *Valor2:* *yicid,xicid:* <val. Inteiros> *yfcid,xfcid:* <val. Inteiro> *Prox:* nil *PrimAdj:* nil *UltAdj:* nil	*CodNodo:* 3 *NomeNodo: Dieta para Cardíaco* *Objetivo: Sim* *Tipo:* *Pergunta:* *Motivo:* *Relacionamento1;* *Valor1:* *Relacionamento2:* *Valor2:* *yicid,xicid:* <val. Inteiros> *yfcid,xfcid:* <val. Inteiro> *Prox:* nil *PrimAdj:* nil *UltAdj:* nil
Registro 4	**Registro 5**	**Registro 6**
CodNodo: 4 *NomeNodo: Dieta para Renal* *Objetivo: Sim* *Tipo:* *Pergunta:* *Motivo:* *Relacionamento1:* *Valor1:* *Relacionamento2:* *Valor2:* *yicid,xicid:* <val. Inteiros> *yfcid,xfcid:* <val. Inteiro> *Prox:* nil *PrimAdj:* nil *UltAdj:* nil	*CodNodo:* 5 *NomeNodo: Dieta para Hepatopata* *Objetivo: Sim* *Tipo:* *Pergunta:* *Motivo:* *Relacionamento1:* *Valor1:* *Relacionamento2:* *Valor2:* *yicid,xicid:* <val. Inteiros> *yfcid,xfcid:* <val. Inteiro> *Prox:* nil *PrimAdj:* nil *UltAdj:* nil	*CodNodo:* 6 *NomeNodo: Trato Intestinal Funcionante* *Objetivo: Não* *Tipo: Univalorada* *Pergunta: O Paciente tem Trato Intestinal Funcionante?* *Motivo:* *Relacionamento1:* *Valor1:* *Relacionamento2:* *Valor2:* *yicid,xicid:* <val. Inteiros> *yfcid,xfcid:* <val. Inteiro> *Prox:* nil *PrimAdj:* nil *UltAdj:* nil

Registro 7	**Registro 8**	**Registro 9**
CodNodo: 7 *NomeNodo: Apetite Preservado* *Objetivo: Não* *Tipo: Univalorada* *Pergunta*: *O apetite do paciente esta preservado?* *Motivo:* *Relacionamento1:* *Valor1:* *Relacionamento2:* *Valor2:* *yicid,xicid:* <val. Inteiros> *yfcid,xfcid:* <val. Inteiro> *Prox:* nil *PrimAdj:* nil *UltAdj:* nil	*CodNodo:* 8 *NomeNodo: Alimentação Parenteral* *Objetivo: Sim* *Tipo* *Pergunta*: *Motivo:* *Relacionamento1:* *Valor1:* *Relacionamento2:* *Valor2:* *yicid,xicid:* <val. Inteiros> *yfcid,xfcid:* <val. Inteiro> *Prox:* nil *PrimAdj:* nil *UltAdj:* nil	*CodNodo:* 9 *NomeNodo: Oral* *Objetivo: Sim* *Tipo:* *Pergunta*: *Motivo:* *Relacionamento1:* *Valor1:* *Relacionamento2:* *Valor2:* *yicid,xicid:* <val. Inteiros> *yfcid,xfcid:* <val. Inteiro> *Prox:* nil *PrimAdj:* nil *UltAdj:* nil

Registro 10
CodNodo: 10 *NomeNodo: Através de Sondas* *Objetivo: Sim* *Tipo:* *Pergunta*: *Motivo:* *Relacionamento1:* *Valor1:* *Relacionamento2:* *Valor2:* *yicid,xicid:* <val. Inteiros> *yfcid,xfcid:* <val. Inteiro> *Prox:* nil *PrimAdj:* nil *UltAdj:* nil

Agora é inserido o delimitador:

Registro 1<Boolean> - delimitador
CodNodo: - *NomeNodo:* *Objetivo:* *Tipo:* *Pergunta*: *Motivo:* *Relacionamento1:* *Valor1:* *Relacionamento2:* *Valor2:* *yicid,xicid:* 0 *yfcid,xfcid:* 0 *Prox:* nil *PrimAdj:* nil *UltAdj:* nil

Gravação das listas de adjacências:

Origem: Nodo 1

Registro 12 – origem	**Registro 13 - adjacente**	**Registro 14 - adjacente**
CodNodo: 1 *NomeNodo: Falência de Órgãos* *Objetivo: Não* *Tipo: Univalorada* *Pergunta: O Paciente tem falência de algum órgão?* *Motivo:* *Relacionamento1:* *Valor1:* *Relacionamento2:* *Valor2:* *yicid,xicid:* 0 *yfcid,xfcid:* 0 *Prox:* nil *PrimAdj:* nil *UltAdj:* nil	*CodNodo:* 2 *NomeNodo:* *Objetivo:* *Tipo:* *Pergunta*: *Motivo:* *Relacionamento1: =* *Valor1: Sim* *Relacionamento2:* *Valor2:* *yicid,xicid:* 0 *yfcid,xfcid:* 0 *Prox:* nil *PrimAdj:* nil *UltAdj:* nil	*CodNodo:* 6 *NomeNodo:* *Objetivo:* *Tipo:* *Pergunta*: *Motivo:* *Relacionamento1: =* *Valor1: não* *Relacionamento2:* *Valor2:* *yicid,xicid:* 0 *yfcid,xfcid:* 0 *Prox:* nil *PrimAdj:* nil *UltAdj:* nil

Registro 15 – delimitador
CodNodo: - *NomeNodo:* *Objetivo:* *Tipo:* *Pergunta*: *Motivo:* *Relacionamento1* *Valor1* *Relacionamento2* *Valor2* *yicid,xicid:* 0 *yfcid,xfcid:* 0 *Prox:* nil *PrimAdj:* nil *UltAdj:* nil

Origem: nodo 2

Registro 16 – origem	**Registro 17 - adjacente**	**Registro 18 - adjacente**
CodNodo: 2 *NomeNodo:* *Objetivo:* *Tipo:* *Pergunta*: *Motivo:* *Relacionamento1:* *Valor1:* *Relacionamento2:* *Valor2:* *yicid,xicid:* 0 *yfcid,xfcid:* 0 *Prox:* nil *PrimAdj:* nil *UltAdj:* nil	*CodNodo:* 3 *NomeNodo:* *Objetivo:* *Tipo:* *Pergunta*: *Motivo:* *Relacionamento1; =* *Valor1: Coração* *Relacionamento2:* *Valor2:* *yicid,xicid:* 0 *yfcid,xfcid:* 0 *Prox:* nil *PrimAdj:* nil *UltAdj:* nil	*CodNodo:* 4 *NomeNodo:* *Objetivo:* *Tipo:* *Pergunta*: *Motivo:* *Relacionamento1: =* *Valor1: Rim* *Relacionamento2:* *Valor2:* *yicid,xicid:* 0 *yfcid,xfcid:* 0 *Prox:* nil *PrimAdj:* nil *UltAdj:* nil

Registro 19 – adjacente	**Registro 20 - delimitador**
CodNodo: 5	*CodNodo:* -

NomeNodo: *Objetivo:* *Tipo:* *Pergunta*: *Motivo:* *Relacionamento1: =* *Valor1: Fígado* *Relacionamento2:* *Valor2:* *yicid,xicid:0* *yfcid,xfcid:0* *Prox:* nil *PrimAdj:* nil *UltAdj:* nil	*NomeNodo:* *Objetivo:* *Tipo:* *Pergunta*: *Motivo:* *Relacionamento1* *Valor1* *Relacionamento2* *Valor2* *yicid,xicid:* 0 *yfcid,xfcid:* 0 *Prox:* nil *PrimAdj:* nil *UltAdj:* nil

Origem nodo 3

Registro 20 – origem	**Registro 22 - delimitador**
CodNodo: 3 *NomeNodo:* *Objetivo:* *Tipo:* *Pergunta*: *Motivo:* *Relacionamento1;* *Valor1:* *Relacionamento2:* *Valor2:* *yicid,xicid:0* *yfcid,xfcid:0* *Prox:* nil *PrimAdj:* nil *UltAdj:* nil	*CodNodo:* - *NomeNodo:* *Objetivo:* *Tipo:* *Pergunta*: *Motivo:* *Relacionamento1* *Valor1* *Relacionamento2* *Valor2* *yicid,xicid:* 0 *yfcid,xfcid:* 0 *Prox:* nil *PrimAdj:* nil *UltAdj:* nil

Origem nodo 4

Registro 23 – origem	**Registro 24 - delimitador**
CodNodo: 4	*CodNodo:* -
NomeNodo:	*NomeNodo:*
Objetivo:	*Objetivo:*
Tipo:	*Tipo:*
Pergunta:	*Pergunta*:
Motivo:	*Motivo:*
Relacionamento1:	*Relacionamento1*
Valor1:	*Valor1*
Relacionamento2:	*Relacionamento2*
Valor2:	*Valor2*
yicid,xicid: 0	*yicid,xicid:* 0
yfcid,xfcid: 0	*yfcid,xfcid:* 0
Prox: nil	*Prox:* nil
PrimAdj: nil	*PrimAdj:* nil
UltAdj: nil	*UltAdj:* nil

Origem nodo 5

Registro 25 - origem	**Registro 26 - delimitador**
CodNodo: 5	*CodNodo:* -
NomeNodo:	*NomeNodo:*
Objetivo:	*Objetivo:*
Tipo:	*Tipo:*
Pergunta:	*Pergunta*:
Motivo:	*Motivo:*
Relacionamento1:	*Relacionamento1*
Valor1:	*Valor1*
Relacionamento2:	*Relacionamento2*
Valor2:	*Valor2*
yicid,xicid: 0	*yicid,xicid:* 0
yfcid,xfcid: 0	*yfcid,xfcid:* 0
Prox: nil	*Prox:* nil
PrimAdj: nil	*PrimAdj:* nil
UltAdj: nil	*UltAdj:* nil

Origem nodo 6

Registro 27 - origem	**Registro 28 - adjacente**	**Registro 29 - adjacente**
CodNodo: 6	*CodNodo:* 7	*CodNodo:* 8
NomeNodo:	*NomeNodo:*	*NomeNodo:*
Objetivo:	*Objetivo:*	*Objetivo:*
Tipo:	*Tipo:*	*Tipo:*
Pergunta:	*Pergunta*:	*Pergunta*:
Motivo:	*Motivo:*	*Motivo:*
Relacionamento1:	*Relacionamento1: =*	*Relacionamento1: =*
Valor1:	*Valor1: Sim*	*Valor1: não*
Relacionamento2:	*Relacionamento2:*	*Relacionamento2:*
Valor2:	*Valor2:*	*Valor2:*
yicid,xicid: 0	*yicid,xicid:* 0	*yicid,xicid:* 0
yfcid,xfcid: 0	*yfcid,xfcid:* 0	*yfcid,xfcid:* 0
Prox: nil	*Prox:* nil	*Prox:* nil
PrimAdj: nil	*PrimAdj:* nil	*PrimAdj:* nil
UltAdj: nil	*UltAdj:* nil	*UltAdj:* nil
Registro 30 - delimitador		

CodNodo: - *NomeNodo:* *Objetivo:* *Tipo:* *Pergunta*: *Motivo:* *Relacionamento1* *Valor1* *Relacionamento2* *Valor2* *yicid,xicid:* 0 *yfcid,xfcid:* 0 *Prox:* nil *PrimAdj:* nil *UltAdj:* nil

Origem nodo 7

Registro 31 - origem	**Registro 32 - adjacente**	**Registro 33 - adjacente**
CodNodo: 7 *NomeNodo:* *Objetivo:* *Tipo:* *Pergunta*: *Motivo:* *Relacionamento1:* *Valor1:* *Relacionamento2:* *Valor2:* *yicid,xicid:* 0 *yfcid,xfcid:* 0 *Prox:* nil *PrimAdj:* nil *UltAdj:* nil	*CodNodo:* 9 *NomeNodo:* *Objetivo:* *Tipo:* *Pergunta*: *Motivo:* *Relacionamento1: =* *Valor1: Sim* *Relacionamento2:* *Valor2:* *yicid,xicid:* 0 *yfcid,xfcid:* 0 *Prox:* nil *PrimAdj:* nil *UltAdj:* nil	*CodNodo:* 10 *NomeNodo:* *Objetivo:* *Tipo:* *Pergunta*: *Motivo:* *Relacionamento1: =* *Valor1: Não* *Relacionamento2:* *Valor2:* *yicid,xicid:* 0 *yfcid,xfcid:* 0 *Prox:* nil *PrimAdj:* nil *UltAdj:* nil

Registro 34 - delimitador
CodNodo: - *NomeNodo:* *Objetivo:* *Tipo:* *Pergunta*: *Motivo:* *Relacionamento1* *Valor1* *Relacionamento2* *Valor2* *yicid,xicid:* 0 *yfcid,xfcid:* 0 *Prox:* nil *PrimAdj:* nil *UltAdj:* nil

Origem nodo 8

Registro 35 - origem	**Registro 36 - delimitador**
CodNodo: 8 *NomeNodo:* *Objetivo:*	*CodNodo:* - *NomeNodo:* *Objetivo:*

Tipo:	*Tipo:*
Pergunta:	*Pergunta*:
Motivo:	*Motivo:*
Relacionamento1:	*Relacionamento1*
Valor1:	*Valor1*
Relacionamento2:	*Relacionamento2*
Valor2:	*Valor2*
yicid,xicid:0	*yicid,xicid:* 0
yfcid,xfcid:0	*yfcid,xfcid:* 0
Prox: nil	*Prox:* nil
PrimAdj: nil	*PrimAdj:* nil
UltAdj: nil	*UltAdj:* nil

Origem nodo 9

Registro 37 - origem	**Registro 38 - delimitador**
CodNodo: 9	*CodNodo:* -
NomeNodo:	*NomeNodo:*
Objetivo:	*Objetivo:*
Tipo:	*Tipo:*
Pergunta:	*Pergunta*:
Motivo:	*Motivo:*
Relacionamento1:	*Relacionamento1*
Valor1:	*Valor1*
Relacionamento2:	*Relacionamento2*
Valor2:	*Valor2*
yicid,xicid: 0	*yicid,xicid:* 0
yfcid,xfcid: 0	*yfcid,xfcid:* 0
Prox: nil	*Prox:* nil
PrimAdj: nil	*PrimAdj:* nil
UltAdj: nil	*UltAdj:* nil

Origem nodo 10

Registro 39 - origem	**Registro 40 - delimitador**
CodNodo: 10	*CodNodo:* -
NomeNodo:	*NomeNodo:*
Objetivo:	*Objetivo:*
Tipo:	*Tipo:*
Pergunta:	*Pergunta*:
Motivo:	*Motivo:*
Relacionamento1:	*Relacionamento1*
Valor1:	*Valor1*
Relacionamento2:	*Relacionamento2*
Valor2:	*Valor2*
yicid,xicid: 0	*yicid,xicid:* 0
yfcid,xfcid: 0	*yfcid,xfcid:* 0
Prox: nil	*Prox:* nil
PrimAdj: nil	*PrimAdj:* nil
UltAdj: nil	*UltAdj:* nil

Capítulo 4

O ambiente de desenvolvimento do SGR

4.1 Introdução

O 'Sistema Gerador de Regras'(SGR) a partir de grafos valorados é um programa elaborado empregando a linguagem de programação "Object Pascal" (Delphi) e tem como objetivo primário ser uma ferramenta de auxílio para que o engenheiro do conhecimento possa construir sistema especialista. No entanto, devido ao fato de ser possível a obtenção de fluxogramas de decisão (que possibilita o desenho dos grafos valorados) em livros textos de medicina, mais especificamente os de 'Bases da Técnica Cirúrgica', o programa permite que qualquer usuário, preferencialmente estudantes de medicina, com conhecimento básico de uso de computadores e interfaces gráficas, possa elaborar bases de conhecimento de sistemas especialistas facilmente, dispensando a presença do engenheiro de conhecimento.

Descrever-se-á neste capitulo a interface gráfica do sistema e a maneira que permite ao usuário, a partir do aplicativo, elaborar bases de conhecimento. Para tal empreitada continuar-se-á desenvolvendo o fluxograma visto no capítulo anterior (melhor dieta para um paciente cirúrgico).

A partir do fluxograma representado na Figura 3.1 pode-se, se for feito para cada nodo que não seja assinalado como 'objetivo', a troca da propriedade 'nome' pela 'pergunta' associada a este nodo, obter-se o fluxograma da figura 4.1 que, quando da implementação do sistema especialista, será o reflexo do motor de inferência. Este fluxograma dá uma idéia de como será a interação homem / máquina quando for ativado o modo de consulta do "shell" de sistema especialista, para isto, basta fazer a navegação através de todos os possíveis caminhos da árvore.

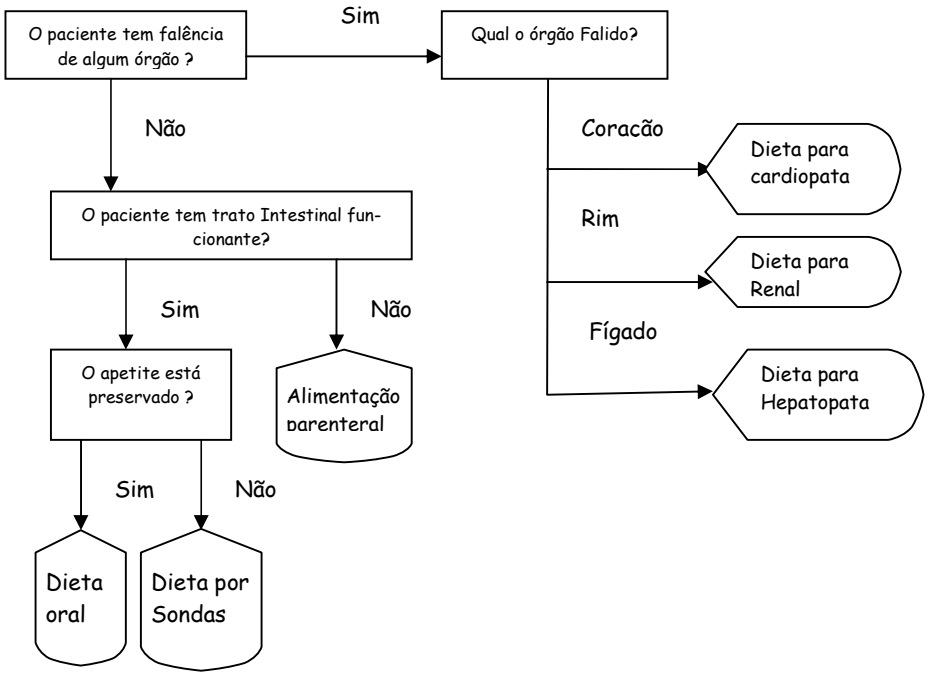


Figura 4. 1– Esquema do motor de inferência quando da implementação do sistema especialista para escolha da melhor dieta para um paciente cirúrgico.

4.2 Tela principal

A tela principal do SGR e todos os componentes, exibidos quando da inicialização do sistema, são mostrados na Figura 4. 4.

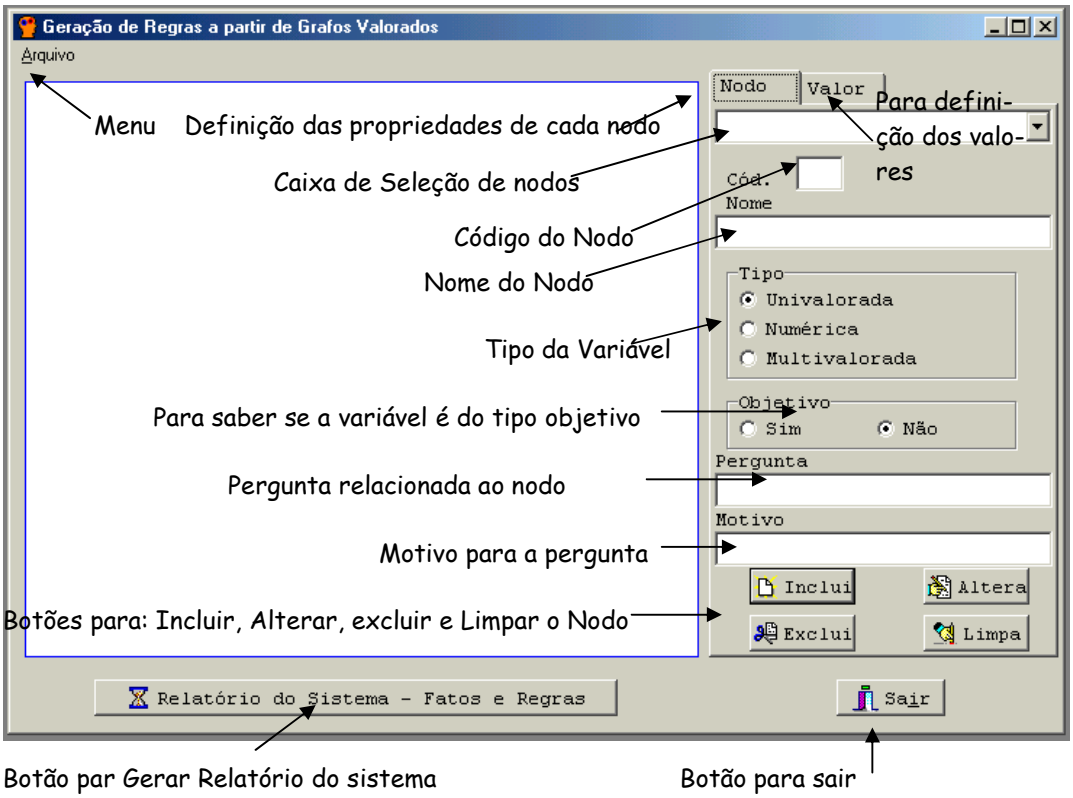


Figura 4. 2- Tela principal do SGR.

Os procedimentos necessários para o desenho do grafo e definição das propriedades dos nodos são descritos a seguir:

4.2.1 Incluir um nodo

Para incluir um nodo, basta preencher o formulário de dados com as propriedades do nodo (Código, Nome, Tipo, se o nodo representa uma variável 'Objetivo' ou não. Se o botão de rádio que define a propriedade 'Objetivo' for marcado como falso (valor 'não') será obrigatório o preenchimento, no formulário, de uma pergunta relacionada aquele nodo. O motivo da formulação da pergunta é de preenchimento opcional).

Após o preenchimento do formulário, pressiona-se o botão 'Inclui'. Após inserção do nodo, o mesmo aparecerá no canto esquerdo superior do desenho do grafo (Figura 4. 3). Para que seja possível a mudança de posição do nodo, basta arrastá-lo para a nova disposição desejada, mantendo o botão esquerdo do mouse pressionado durante o processo.

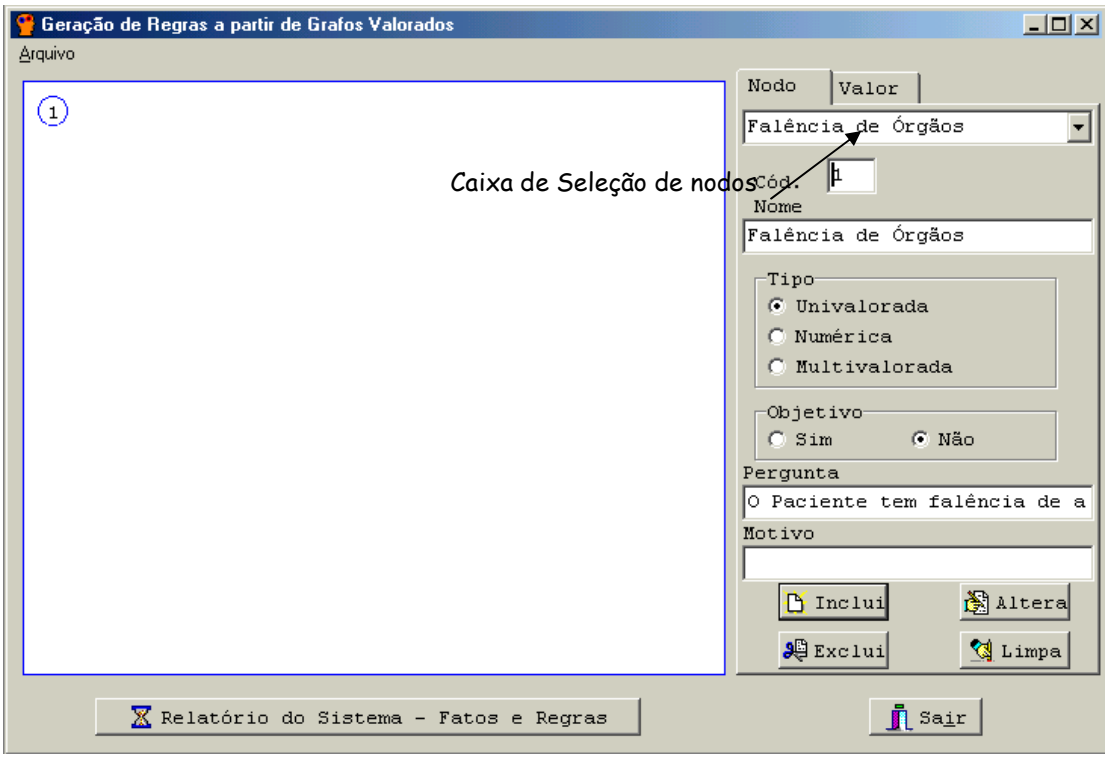


Figura 4. 3 – Inclusão do primeiro nodo no grafo

4.2.2 Selecionar um nodo

Há duas maneiras de se selecionar um nodo na tela de desenho. A primeira é pressionando o botão esquerdo do mouse sobre o nodo desejado na tela. Automaticamente, o formulário de dados sobre o nodo será atualizado com os dados relativos ao nó selecionado.

A segunda maneira é a partir da escolha de um item da caixa de seleção (nome do nodo desejado), que está localizada no formulário de dados (Figura 4.3).

4.2.3 Movendo um nodo na tela

Para movimentação de um determinado nodo, seleciona-se o nodo com o mouse e arrasta-o para a nova posição. (Obs.: Para arrastar conserva-se o botão esquerdo do mouse pressionado até chegar na posição desejada)

4.2.4 Alterando dados de um nodo

Para alterar os dados de um nodo, basta selecioná-lo e efetuar modificações nos valores constantes no formulário. Após colocação dos novos dados, pressiona-se o botão 'Altera' para validação das mudanças.

4.2.5 Excluindo um nodo

Para exclusão, seleciona-se o nodo desejado e pressiona-se o botão 'Exclui'. Todas as suas ligações com os nodos vizinhos serão automaticamente eliminadas.

4.2.6 Limpando o formulário de dados de um nodo

Se o usuário desejar limpar o formulário de entrada, basta pressionar o botão que apresenta o rótulo 'Limpa'.

4.2.7 Incluído um nodo 'Objetivo'

Quando o usuário deseja incluir um nodo que apresente a propriedade 'objetivo' definida como verdadeira ('Sim'), as caixas de textos contendo as opções: pergunta e a de motivo da pergunta, bem como os botões de seleção para escolha do tipo de nodo ficarão invisíveis (um atributo do tipo objetivo é uma folha, e, portanto, não permite formulação de perguntas). Para diferenciação com outras variáveis, que não são objetivos do sistema, o desenho do nodo incluído ostentará o formato de um quadrado (Figura 4. 4).

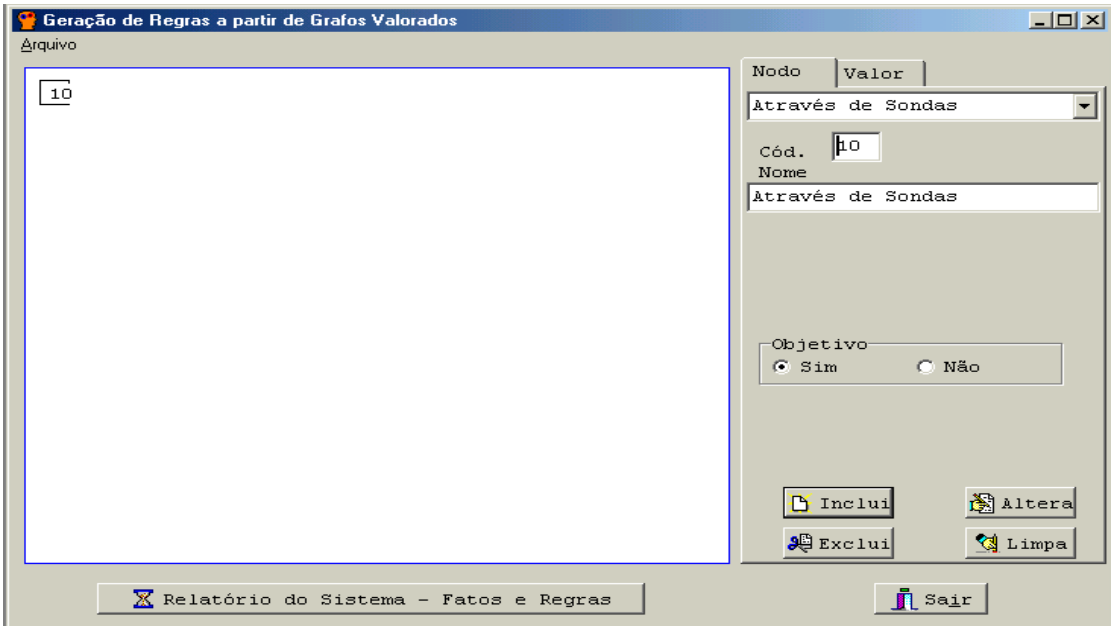


Figura 4. 4 – Inclusão de uma variável tipo Objetivo

4.2.8 Atribuindo sinal de relacionamento e valores entre os nodos (ligando nodos)

Clica-se no formulário de dados para atribuição de valores entre os nodos (aba 'Valor'). Seleciona-se na caixa de seleção origem e na caixa de seleção destino o nodo de origem e destino, respectivamente. Coloca-se o sinal de relacionamento (= ou <>) e o valor do nodo origem que permitem gerar o nodo destino. Se esse relacionamento e valor forem os mesmos de ida e volta (relacionamento em mão dupla) seleciona-se a opção 'Sim' na caixa de seleção 'Mão Dupla' (Figura 4. 5).

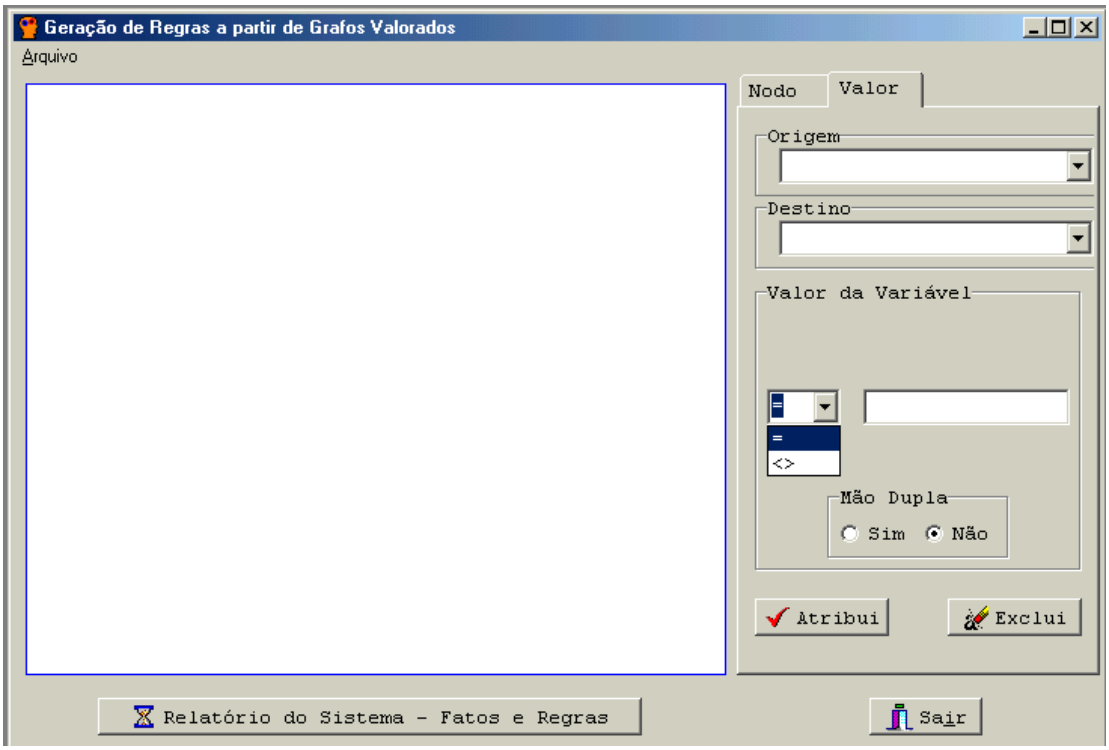


Figura 4. 5 – Interface para atribuição de valor que liga um nodo origem a um destino e os sinais de atribuição "=" e "<>" evidenciados.

Quando o nodo origem é do tipo 'Numérico', a caixa de seleção permite a opção de escolha de um único número ou de um intervalo de valores.

Quando a opção é por um valor numérico único e não um intervalo só será permitido que o sinal de atribuição assuma valores '=' ou '<>' (Figura 4. 6).

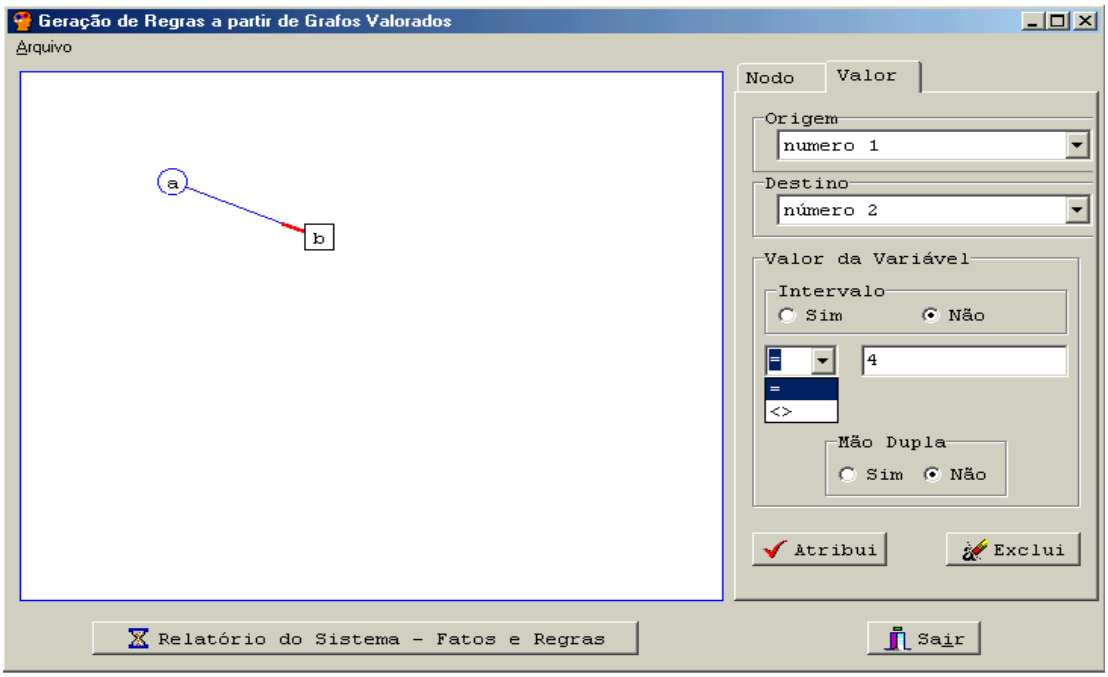


Figura 4. 6 – Opções de sinal de atribuição quando o nodo origem é do tipo numérico. Observe que se pretende um número e não um intervalo numérico.

Quando a opção é por um intervalo numérico duas caixas de seleção, para escolha dos sinais de atribuição e para definição dos valores que comporão o intervalo estarão presentes, para uma delas será introduzido o limite inferior do intervalo e na outra o limite superior (Figuras 4. 7 e 4. 8).

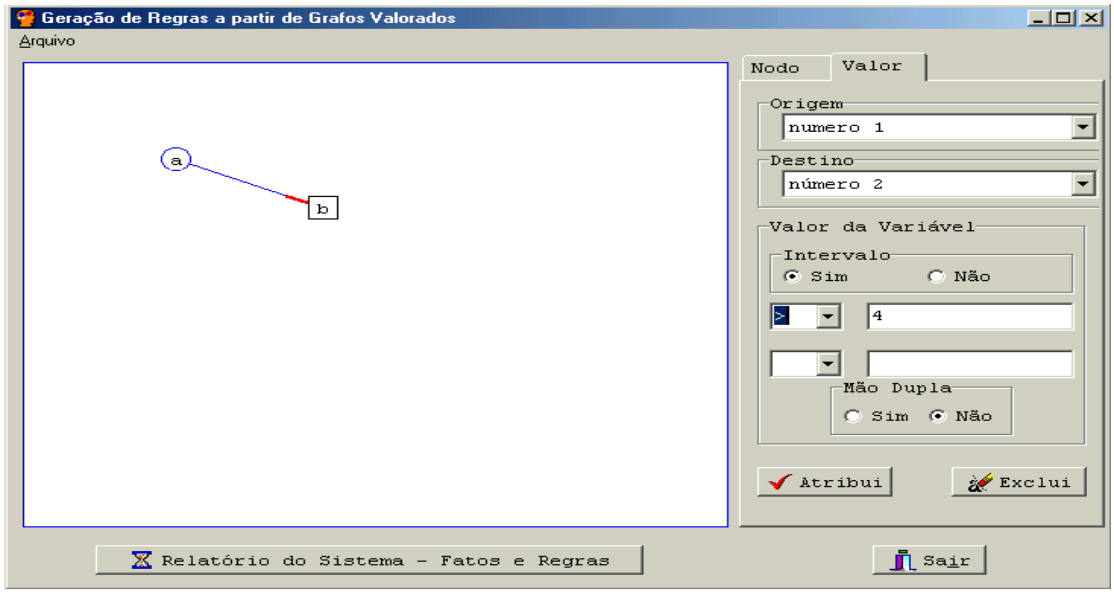


Figura 4. 7 – Definição de valores para um intervalo numérico entre um nodo origem e um nodo destino. O sinal de atribuição selecionado define o limite inferir do intervalo ('>4').

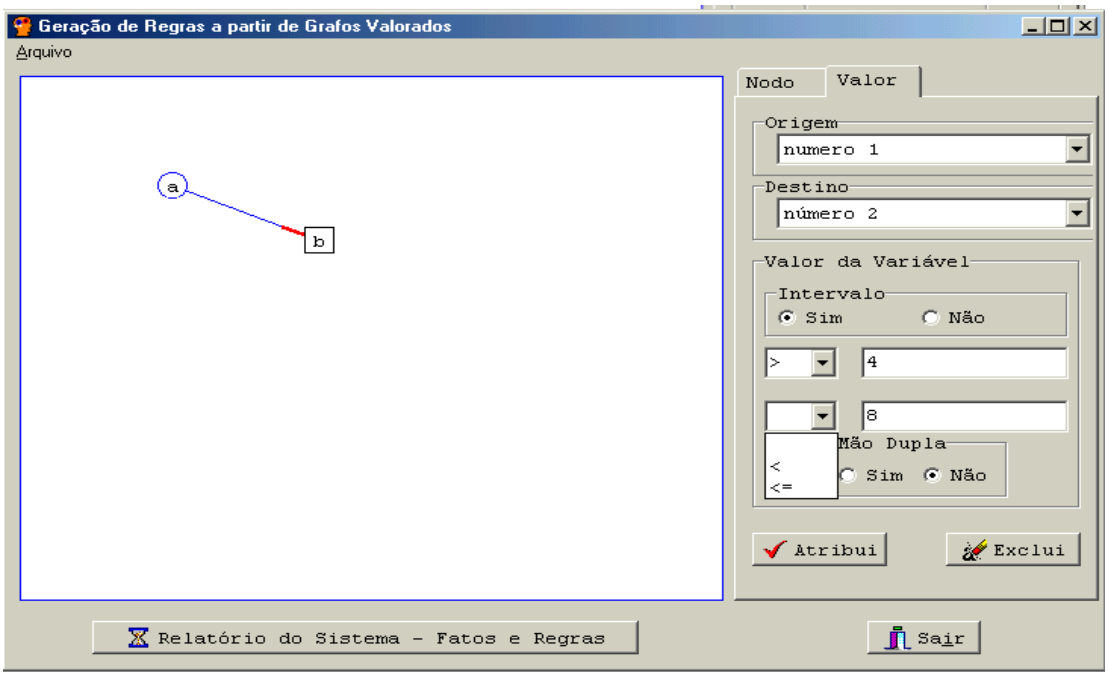


Figura 4. 8 - Definição de valores para um intervalo numérico entre um nodo origem e um nodo destino. Em evidencia os possíveis valores que o sinal de atribuição poderão assumir para definição do limite superior do intervalo (‘ ‘, ‘<’, ‘<=’).

Quando o nodo de origem é do tipo numérico e existe tentativa de introdução de um valor que não seja um número, uma ‘,’ ou um ‘.’, uma caixa de mensagem é exibida alertando para tal situação e nenhuma entrada será permitida (Figura 4. 9).

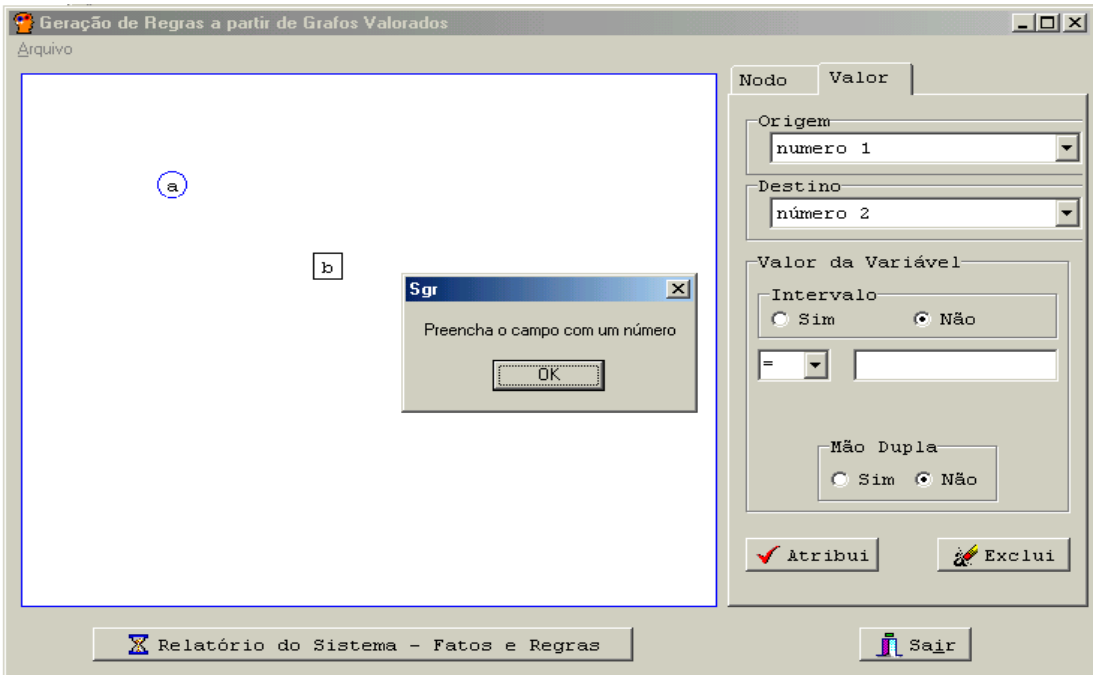


Figura 4. 9 – Tentativa de introdução de um valor não numérico para um nodo origem numérico.

Quando terminado o preenchimento dos valores que relacionam um nodo ao outro o botão 'Atribui' é pressionado e essas dois nodos estarão conectados. Observe no desenho da tela que a ligação entre dois nós é feita por uma linha azul com uma ou duas das extremidades vermelhas. A linha vermelha no final da ligação indica que o nodo que está recebendo essa parte vermelha é destino do outro (Figura 4. 10). Quando é 'Mão-Dupla', a linha vermelha aparecerá nas duas extremidades.

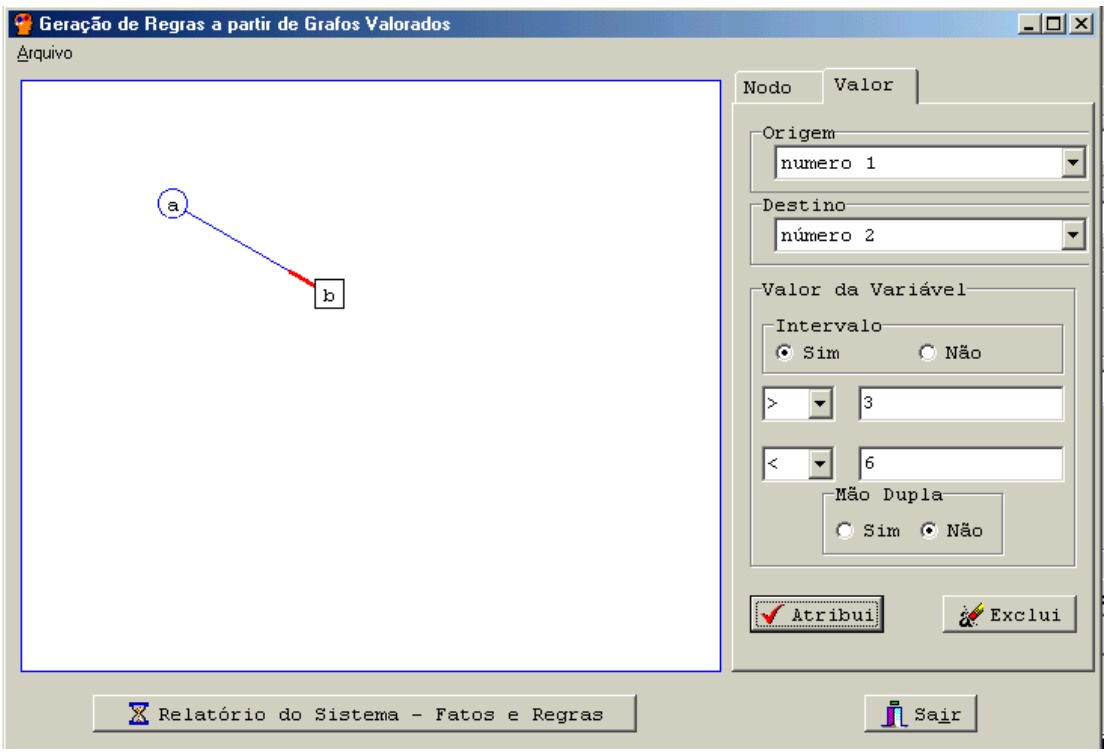


Figura 4. 10 – Sinais de atribuição e valores entre os nodos, cuja tradução seria o número 2 só aparece se o número 1 for maior que 3 e menor que 6.

4.2.9 Consultando sinal de relacionamento e valor entre dois nodos

Clica-se na aba 'Valor'. Seleciona-se o nodo origem e o nodo destino nas caixas combo. O(s) sinal(is) de relacionamento e valor entre eles serão exibidos automaticamente nos campos 'Relacionamento' e 'Valor'.

4.2.10 Excluindo o valor e o sinal de relacionamento entre nodos (excluindo ligações)

Clica-se na aba 'Valor'. Seleciona-se o nodo origem e o destino e pressiona-se o botão 'Exclui'.

4.3 Definido o objetivo do sistema

Quando se elabora um fluxograma de decisão é por que se quer atingir um objetivo (um diagnóstico, a escolha de um tipo de cirurgia, tipo de exames, etc).

O objetivo tem um nome (diagnóstico, exame, medicamento, dieta, etc.) e um valor (Diagnóstico = apendicite, coleciste, colangite, diverticulite, etc.).

Um fluxograma que só permite uma decisão por vez é constituído por um objetivo do tipo univalorado (Ex. Se o paciente tem 'falência de órgão' e este órgão é o 'fígado' a variável 'Dieta' para este paciente é especifica para paciente 'hepatopata' – Sem aminoácidos aromáticos, contendo aminoácidos de cadeia aromática).

Um fluxograma que permite que mais de um valor seja obtido é constituído de variável multivalorada (Ex. O paciente que tem 'antecedentes' de vômitos e diarréia e apresenta 'sintomatologia' para sede, mucosas secas, pulso fino, hipotensão arterial e oligúria pode apresentar a variável do tipo 'objetivo' com os valores = desidratação, hipocloremia, hiponatremia, hipopotassemia ao mesmo tempo).

A variável do tipo 'Objetivo' só será definida após desenho completo do grafo e, dependendo da vontade do implementador da base de conhecimentos, poderá ter suas propriedades modificadas para cada encaminhamento (todas as vezes que o botão 'Relatório do Sistema – Fatos e Regras' for clicado).

Quando o botão 'Relatório do Sistema – Fatos e Regras' é clicado, existe necessidade de definição das propriedades da variável objetivo, para tal, é necessário responder as duas caixas de entrada de texto que serão exibidas seqüencialmente (Figuras 4. 11 e 4. 12).

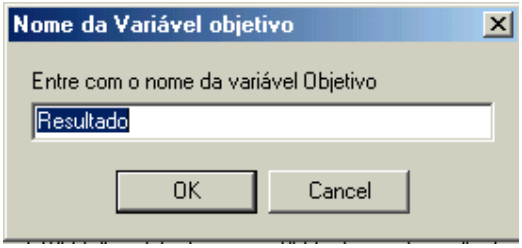


Figura 4. 11 - Tela para atribuição do nome da variável objetivo do sistema.

Para construção da base de conhecimento 'Melhor Dieta para um Paciente Cirúrgico', não será aceito o valor "default" – 'Resultado', e em função disso, altera-se o nome para 'Dieta'. Para que a alteração seja validada clica-se o botão com o rótulo 'OK'. Se o

botão contendo o rótulo ‘Cancel’ for pressionado, o sistema manterá o “default”, ou seja, o nome da variável objetivo aceitará o valor padrão ‘Resultado’.

Definido o nome do atributo objetivo, nova janela (Input Box) é exibida para definição do tipo de variável que se pretende para o sistema: Univalorada ou Multivalorada (Figura 4. 12).

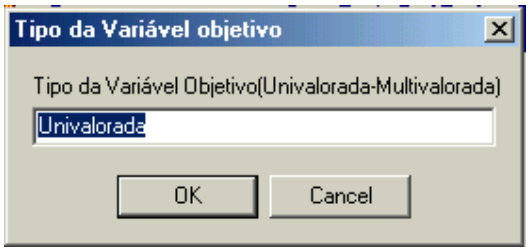


Figura 4. 12 - Tela para definição do tipo da variável Objetivo.

No exemplo de sistema, que está sendo desenvolvido, só é permitido um tipo de dieta por vez e o “default” é aceito – ‘Univalorada’.

4.4 Relatório do sistema

Após desenho do grafo e preenchimento dos atributos dos nodos (Veja figura 4. 13 e tabelas 4.1 e 4. 2), o relatório, contendo as variáveis e regras para alimentação do sistema especialista só aguarda o desejo do usuário para ser exibido.

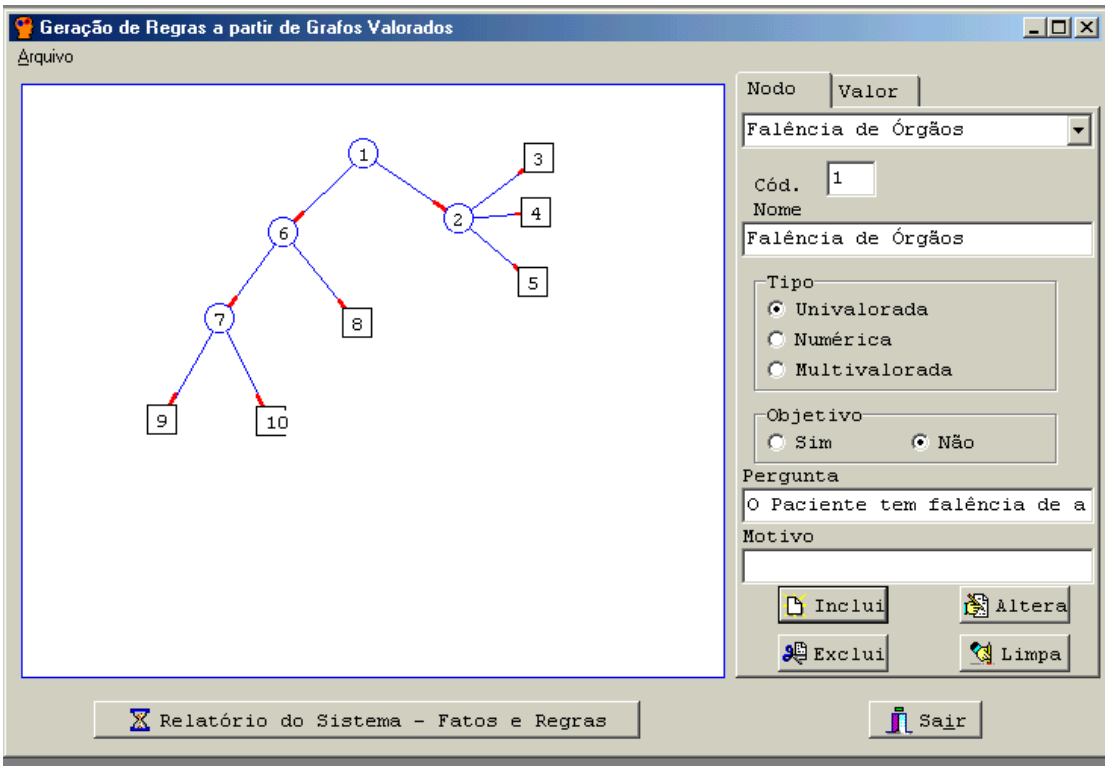


Figura 4. 13 – Desenho completo do Grafo

Cod	Nome	Tipo	Objetivo	Pergunta	Motivo
1	Falência de Órgãos	Univalorada	Não	O Paciente tem falência de algum órgão?	Xxxxxx
2	Condição Especial	Univalorada	Não	Qual o órgão Falido?	Xxxxxx
3	Dieta para Cardíaco		Sim		Xxxxxx
4	Dieta para Renal		Sim		Xxxxxx
5	Dieta para Hepatopata		Sim		Xxxxxx
6	Trato Intestinal Funcionante	Univalorada	Não	O Paciente tem Trato Intestinal Funcionante?	Xxxxxx
7	Apetite Preservado	Univalorada	Não	O apetite do paciente esta preservado?	Xxxxxx
8	Alimentação Parenteral		Sim		Xxxxxx
9	Oral		Sim		Xxxxxx
10	Através de Sondas		Sim		Xxxxxx

Tabela 4. 1 – Atributos dos nodos

Origem/Destino	Relacionamento	Valor	Mão Dupla
Falência de Órgão/Condição Especial	=	Sim	Não
Condição Especial/Dieta para Cardíaco	=	Coração	Não
Condição Especial/Dieta para Renal	=	Rim	Não
Condição Especial/Dieta para Hepatopata	=	Fígado	Não
Falência de Órgão/Trato Intestinal Funcionante	=	Não	Não

Trato Intestinal Funcionante/Apetite Preservado	=	Sim	Não
Trato Intestinal Funcionante/Alimentação Parenteral	=	Não	Não
Apetite Preservado/Oral	=	Sim	Não
Apetite Preservado/Através de Sondas	=	Não	Não

Tabela 4. 2 – Relacionamento e valor entre os nodos origem e destino.

4.4.1 Relatório do Sistema – Fatos e Regras

Após definição do nome e do tipo da variável objetivo (vide definindo objetivo do sistema) clica-se o botão 'Relatório do Sistema – Fatos e Regras' e uma nova tela de formulário é exibida contendo as variáveis (Figura 4.14) e as regras (Figura 4. 15) que permitirão a implementação do sistema especialista.

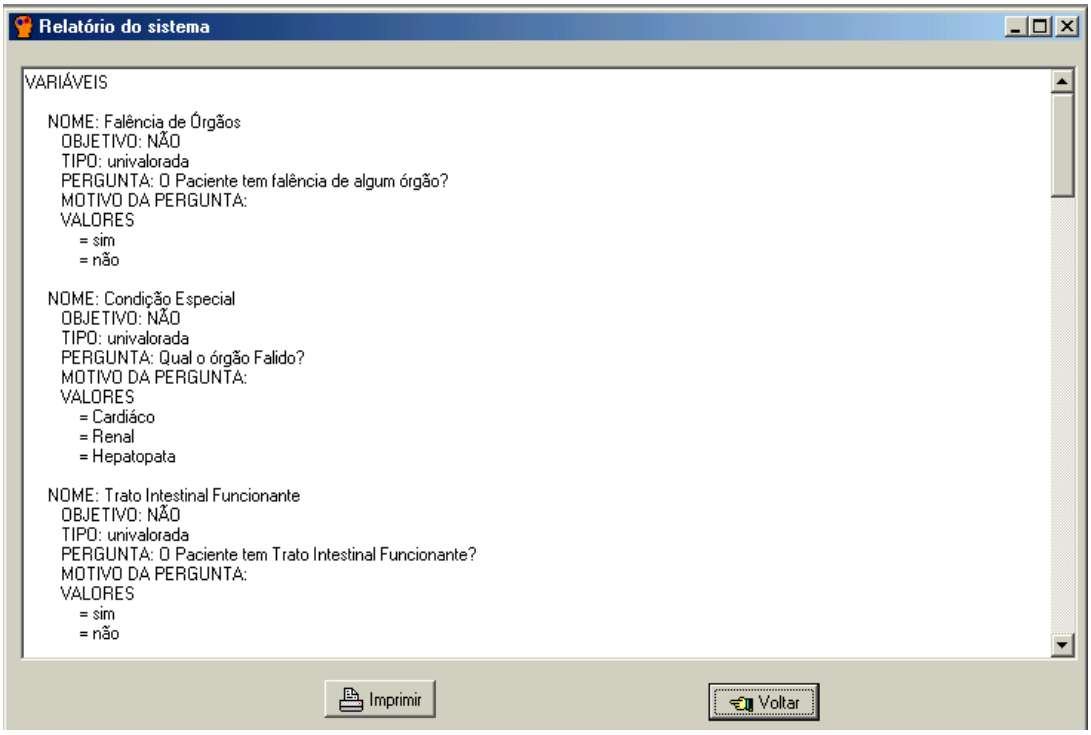


Figura 4. 14 – Relatório do Sistema – Visão parcial das variáveis que comporão o sistema especialista.

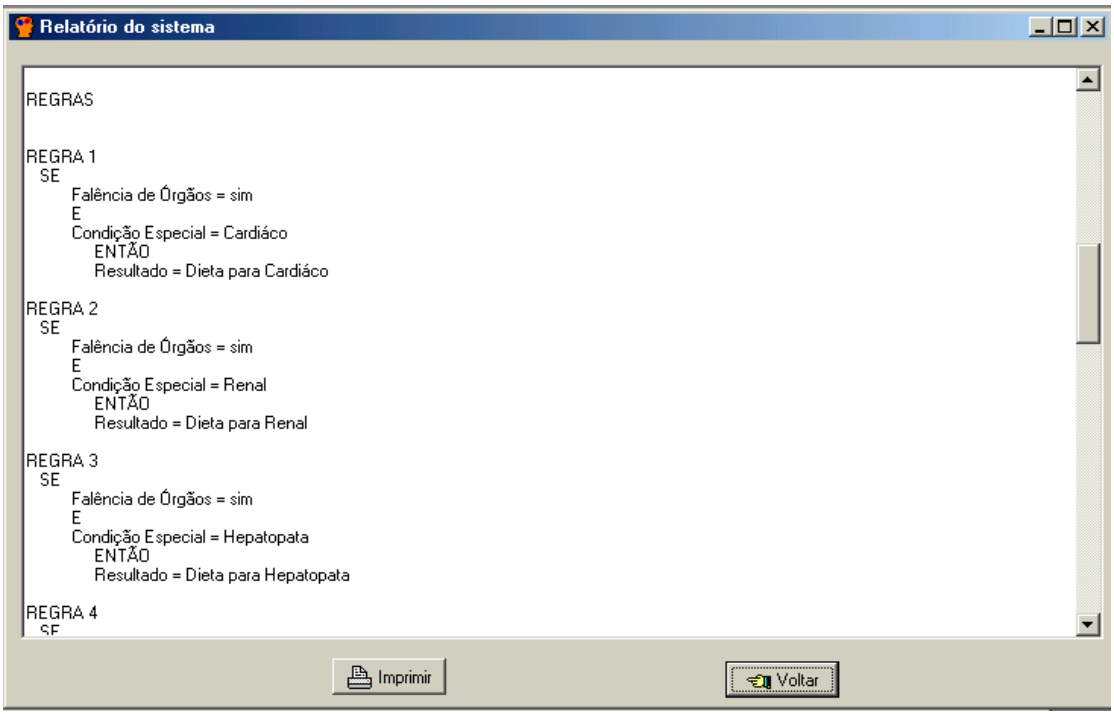


Figura 4. 15 – Relatório do Sistema – Visão parcial das regras.

O Relatório gerado, para o sistema exemplificado, será:

VARIÁVEIS

NOME: Falência de Órgãos
OBJETIVO: NÃO
TIPO: univalorada
PERGUNTA: O Paciente tem falência de algum órgão?
MOTIVO DA PERGUNTA:
VALORES
= Sim
= Não

NOME: Condição Especial
OBJETIVO: NÃO
TIPO: univalorada
PERGUNTA: Qual o órgão Falido?
MOTIVO DA PERGUNTA:
VALORES
= Coração
= Rim
= Fígado

NOME: Trato Intestinal Funcionante
OBJETIVO: NÃO
TIPO: univalorada
PERGUNTA: O Paciente tem Trato Intestinal Funcionante?
MOTIVO DA PERGUNTA:
VALORES
= Sim
= Não

NOME: Apetite Preservado
OBJETIVO: NÃO
TIPO: univalorada
PERGUNTA: O apetite do paciente esta preservado?
MOTIVO DA PERGUNTA:
VALORES
= Sim
= Não

```
   NOME: Dieta
     OBJETIVO: SIM
     Tipo: Univalorada
     VALORES
        = Dieta para Cardíaco
        = Dieta para Renal
        = Dieta para Hepatopata
        = Alimentação Parenteral
        = Oral
        = Através de Sondas


REGRAS


REGRA 1
  SE
       Falência de Órgãos = Sim
         E
       Condição Especial = Coração
          ENTÃO
          Dieta = Dieta para Cardíaco

REGRA 2
  SE
       Falência de Órgãos = Sim
         E
       Condição Especial = Rim
          ENTÃO
          Dieta = Dieta para Renal

REGRA 3
  SE
       Falência de Órgãos = Sim
         E
       Condição Especial = Fígado
          ENTÃO
          Dieta = Dieta para Hepatopata

REGRA 4
  SE
       Falência de Órgãos = Não
         E
       Trato Intestinal Funcionante = Não
          ENTÃO
          Dieta = Alimentação Parenteral

REGRA 5
  SE
       Falência de Órgãos = Não
         E
       Trato Intestinal Funcionante = Sim
         E
       Apetite Preservado = Sim
          ENTÃO
          Dieta = Oral

REGRA 6
  SE
       Falência de Órgãos = Não
         E
       Trato Intestinal Funcionante = Sim
         E
       Apetite Preservado = Não
          ENTÃO
          Dieta = Através de Sondas

REGRA 7
  SE
       Condição Especial = Coração
          ENTÃO
          Dieta = Dieta para Cardíaco
```

```
REGRA 8
  SE
       Condição Especial = Rim
          ENTÃO
          Dieta = Dieta para Renal

REGRA 9
  SE
       Condição Especial = Fígado
          ENTÃO
          Dieta = Dieta para Hepatopata

REGRA 10
  SE
       Trato Intestinal Funcionante = Não
          ENTÃO
          Dieta = Alimentação Parenteral

REGRA 11
  SE
       Trato Intestinal Funcionante = Sim
        E
       Apetite Preservado = Sim
          ENTÃO
          Dieta = Oral

REGRA 12
  SE
       Trato Intestinal Funcionante = Sim
        E
       Apetite Preservado = Não
          ENTÃO
          Dieta = Através de Sondas

REGRA 13
  SE
       Apetite Preservado = Sim
          ENTÃO
          Dieta = Oral

REGRA 14
  SE
       Apetite Preservado = Não
          ENTÃO
          Dieta = Através de Sondas
```

4.4.2 Impressão do relatório final

Para impressão do relatório gerado pelo sistema clica-se no botão com o rótulo 'Imprimir'.

Figura 4. 16 – Botão para imprimir relatório.

Após escolha da sua Impressora clica-se no botão 'OK' (Figura 4. 17).

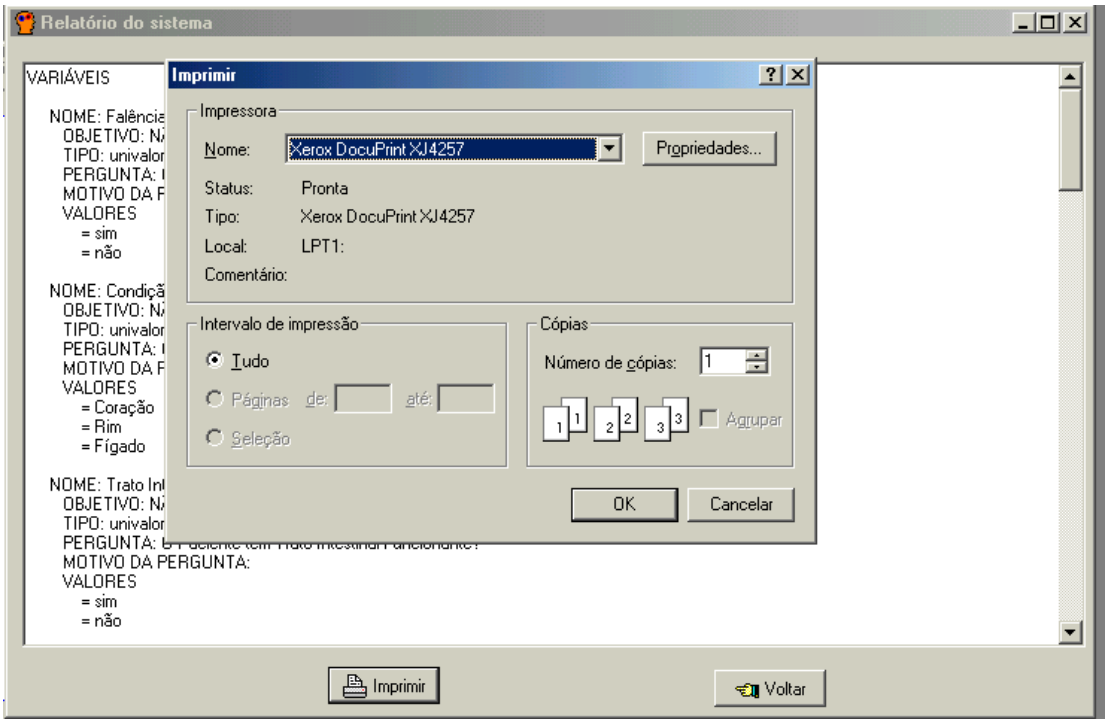


Figura 4. 17 – Tela para escolha da impressora.

4.4.3 Voltar à tela principal

Para retornar a tela principal clica-se no botão com o rótulo ‘Voltar’ (Figura 4. 18)

Figura 4. 18 – Botão para retornar a Tela Principal.

4.5 Saída e fechamento do programa

Para sair do programa, são possíveis as opções:

- Clica-se no botão ‘Sair’ (Figura 4. 19), ou;

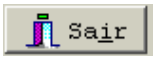


Figura 4. 19 – Botão Para sair do Programa

- Clica-se no botão ‘X’ da barra de título (Figura 4. 20), ou;

Figura 4. 20 – Para sair clique no X da barra de título

- No menu principal clica-se em ‘Arquivo’ e finalmente em ‘Salvar’ (Figura 4. 21).

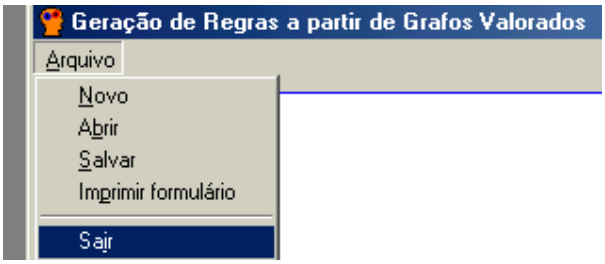


Figura 4. 21 – Opção de saída através do Menu.

Qualquer que seja a opção de fechamento é estabelecido diálogo para que seja possível a operação de salvamento do grafo (Figura 4. 22).

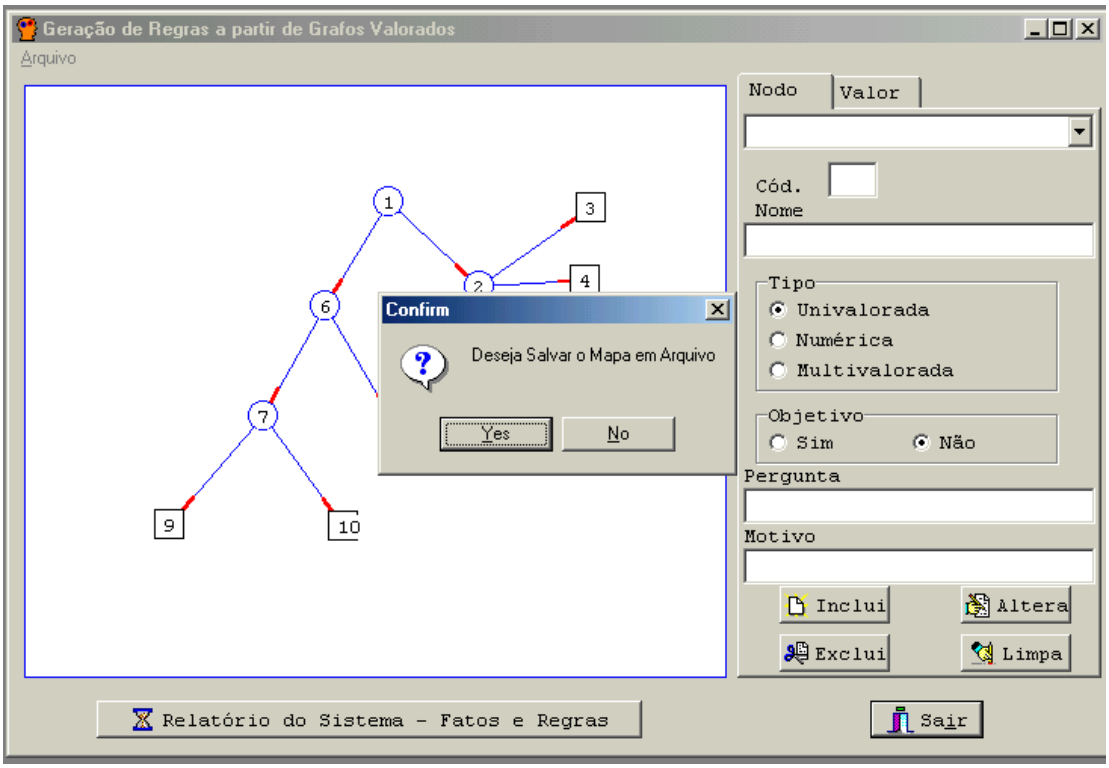


Figura 4. 22 – Confirmação de salvamento do grafo

Se o desejo de salvamento for afirmativo novo dialogo é estabelecido para escolha do nome do arquivo e do diretório (Figura 4. 23).

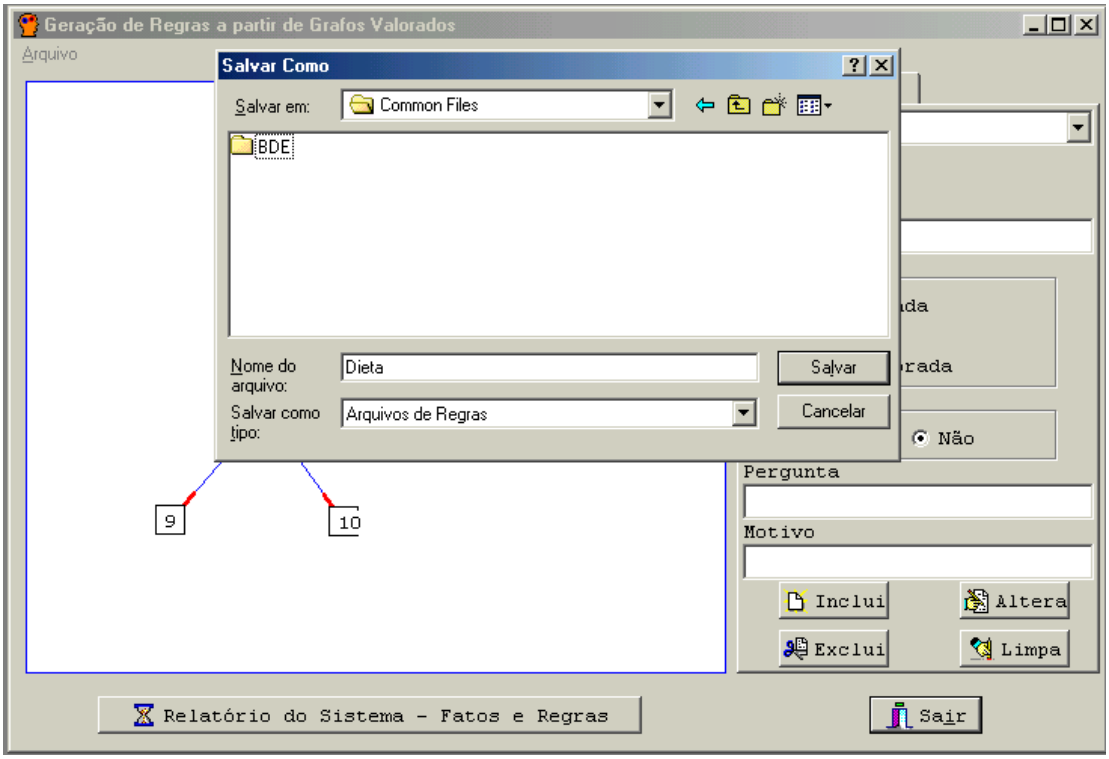


Figura 4. 23 – Escolha do nome do arquivo e o diretório para salvamento

A qualquer momento o trabalho pode ser salvo a partir da opção de Menu ‘Arquivo’ – ‘Salvar’.

4.6 Outras opções

4.6.1 Abrir um arquivo

Para abrir um arquivo previamente criado segue-se a seqüência ‘Arquivo’ / ‘Abrir’ e seleciona-se um arquivo com extensão btc (Figura 4. 24).

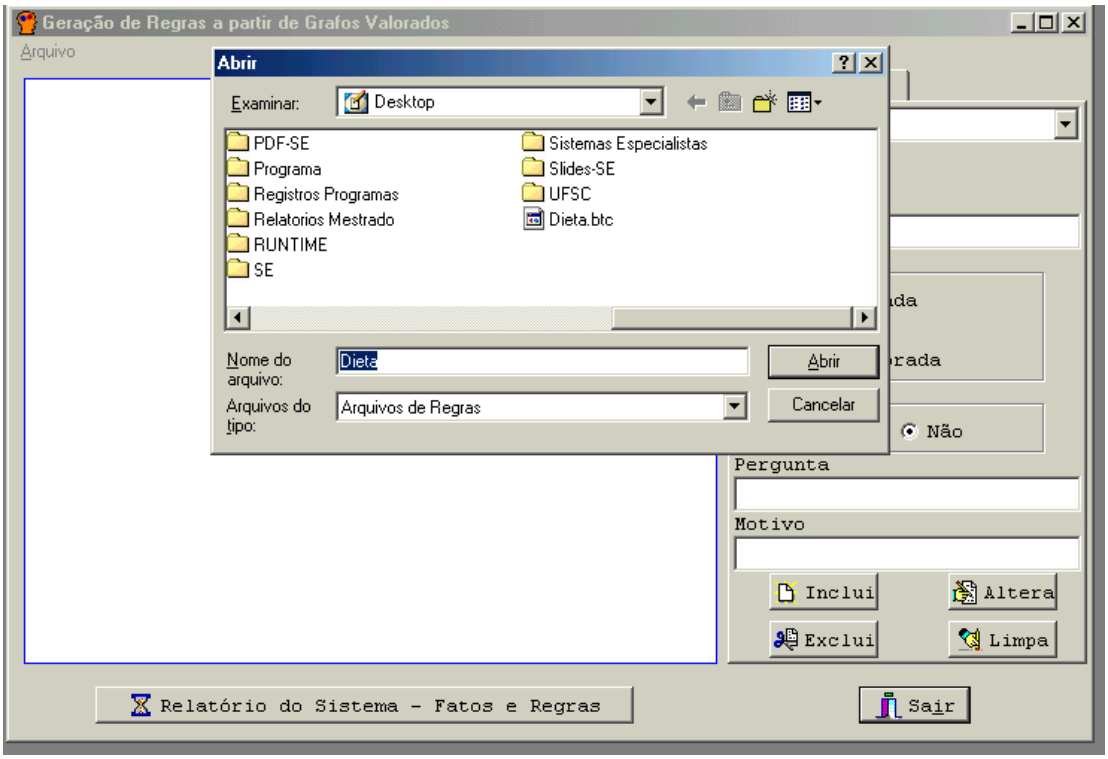


Figura 4. 24 – Abertura de arquivo.

4.6.2 Criação de um novo arquivo

Seguindo a seqüência ‘Arquivo’ / ‘Novo’ salve-se o grafo anterior e um novo é inicializado.

4.6.3 Impressão do formulário com o grafo.

Para que seja possível a impressão do formulário visível na área de trabalho da tela do computador e do desenho do grafo, clica-se em ‘Arquivo’ e ‘Imprimir formulário’.

Capítulo 5

Construção de Sistemas Especialistas

5.1 Introdução

Na avaliação de Biczyc, o conhecimento médico pode ser obtido e corretamente extraído a partir de três fontes [Biczyc et all 1993]:

- Fontes teóricas como, por exemplo: livros médicos, manuais e literatura em geral.
- Fontes empíricas, tal como banco de dados de pacientes.
- Especialista médico.

O papel do engenheiro do conhecimento é gerar uma base de dados a partir de uma fonte bruta, denominada banco ou base de conhecimento, que serve de esteio para a construção de sistema especialista em um determinado domínio do saber.

A base de conhecimento pode ser armazenada sob um ou mais formatos: regras de produção, scripts, frames, redes semânticas, etc.

Neste capítulo, continuar-se-á desenvolvendo o sistema especialista para escolha da melhor dieta do paciente cirúrgico, cujas regras (regras de produção) e variáveis foram gerados pelo SGR e que são a base para montagem do banco de conhecimento.

O uso de SGR , como se viu anteriormente, torna facultativa a presença do engenheiro do conhecimento, sendo pré-requisito básico para montagem do sistema especialista apenas a necessidade de um fluxograma de decisão obtido de um livro texto, elaborado pelo professor ou mesmo montado pelo aluno a partir do entendimento do assunto abordado em aula.

Porém, o fluxograma não é a única forma de representação do conhecimento. Quando usado em um domínio de problema muito grande, o fluxograma provavelmente ficará enorme, porque o número de possíveis sucessões de situações a ser considerado será abissal.

Foi construído e será mostrado, também, neste capitulo, um sistema especialista mais complexo que servirá para 'Diagnóstico de distúrbios hidroeletrolíticos', onde o fluxograma de decisão, por se só, não é adequado para representar todo o conhecimento médico envolvido. Parte das regras, que formam a base de conhecimento deste segundo sis-

tema, foi construída com o auxílio de técnicas de engenharia do conhecimento, e outra parte, com o 'Sistema Gerador de Regras'.

Para exemplificação da funcionalidade e aplicabilidade dos sistemas especialistas no ensino da disciplina 'Bases da Técnica Cirúrgica', após construção das bases de conhecimentos as mesmas foram montadas em um "shell", o 'Expert Sinta', para permitir ao aluno um aprendizado interativo dos assuntos abordados em aula. Todos os passos necessários para construção dos sistemas serão descritos adiante e ilustrados com figuras.

5.2 – Objetivo

O ensino da Medicina é atividade desafiadora, na medida que a lógica empregada para diagnosticar uma doença nem sempre se resume a um algoritmo simples: "*se tais sintomas e sinais então tal doença*". Devido à natureza probabilística nem todos os achados esperados estão presentes e mais de uma doença poderão ser resultantes da associação de um dado conjunto de sintomas e sinais.

O objetivo deste capítulo é mostrar a construção de dois sistemas especialistas, onde alguns desses aspectos são evidenciados.

Para o primeiro sistema, o aluno, desde que tenha domínio básico de uso de computadores e interfaces gráficas, poderá construir sozinho um pequeno sistema especialista utilizando o SGR e o "shell" Expert Sinta. Este sistema foi montado a partir do relatório gerado no SGR para o fluxograma que permite decisão para escolha da melhor dieta para um paciente cirúrgico, cujo desenvolvimento esta documentado nos capítulos anteriores.

Para o segundo sistema, além da ferramenta empregada para o primeiro sistema (SGR), fez-se necessário o emprego de técnicas de engenharia do conhecimento para elaboração de algumas regras. Este segundo módulo tratou do 'Diagnóstico de distúrbios hidroeletrolíticos'.

O uso de sistemas especialistas possibilita ao aluno de 'Bases da Técnica Cirúrgica' uma opção a mais para fixação de assuntos abordados em sala de aula. Sem necessidade de decorar (no sentido de memorização) permite treinamento para desenvolvimento do raciocínio empregado em atividades decisórias na pratica médica (elaboração de diagnósticos, prescrição de medicamentos, solicitações de exames, dentre outros). O aprendizado é alcançado a partir da interação homem / máquina: o aluno tomará decisões a partir de perguntas formuladas pelo computador.

5.3 – Relevância

Para que se possa avaliar a relevância do tema, tomar-se-á, como exemplo, o manuseio de distúrbios hidroeletrolíticos, um dos assuntos abordados na disciplina 'Bases da Técnica Cirúrgica', que será objeto do segundo sistema especialista.

O manuseio dos distúrbios hidroeletrolíticos é tarefa nem sempre agradável para a maioria dos médicos, sendo quase sempre feito de maneira empírica, acarretando, quando perpetrada de modo incorreto, trabalho extra ao rim do paciente, que além de realizar sua tarefa normal de homeostasia, terá que reagir frente ao uso inadequado de soluções no que se refere a sua qualidade e quantidade.

As razões pelas quais, no manuseio dos distúrbios hidroeletrolíticos, haja prática inadequada por parte de alguns profissionais e estudantes de Medicina são, dentre outras, as que se seguem:

- Desconhecimento ou preguiça para realizar cálculos matemáticos, por mais simples que sejam.
- Necessidade de conhecimento de bioquímica.
- Pobreza de publicações literárias com exercícios práticos contendo simulação de casos clínicos de pacientes com distúrbios hidroeletrolíticos.
- Necessidade de processos heurísticos para interpretação de exames laboratoriais de dosagens de eletrólitos. Exemplo: O sódio está baixo por diminuição do conteúdo ou por hiperhidratação?
- Possibilidade de opção por varias soluções contendo eletrólitos em diferentes concentrações para correção de um dado distúrbio.
- Vários métodos utilizados para correção de uma dada situação. Pode-se, por exemplo, tratar desidratação com o conhecimento do balanço hidroeletrolítico, com a dosagem laboratorial do sódio, através da medida do hematócrito, tateando-se o déficit empiricamente dentre outros métodos.
- Influência das condições orgânicas no estado de hidratação do paciente: estado nutricional, idade, função cardiovascular, função renal, dentre outros.

5.4 – O shell

Como ferramenta computacional que utiliza técnicas de inteligência artificial para geração automática de sistemas especialistas, optou-se pelo EXPERT SINTA.

O Expert Sinta é um "shell" implementado na linguagem de programação orientada a objetos (Object Pascal - Delphi), desenvolvido pela Universidade do Ceará através do grupo SINTA (Sistemas Inteligentes Aplicados), cuja característica principal é a simplificação do trabalho de implementação de sistemas especialistas.

O Expert Sinta emprega um modelo de representação do conhecimento baseado em regras de produção e probabilidades. Outras características do software são:

- O uso de uma máquina de inferência compartilhada.
- A construção automática de telas e menus.
- O tratamento probabilístico das regras de produção.
- A utilização de explicações sensíveis ao contexto da base de conhecimento modelada.

Uma das facilidades para justificativa do seu uso é a fácil manutenção que o usuário terá com a ferramenta, como por exemplo, a inclusão, alteração e exclusão de novas regras. Além disso, o usuário não necessita de conhecimentos em programação.

A utilização de uma ferramenta do tipo "shell" simplifica a construção de sistemas especialistas e facilita enormemente sua possível manutenção. Essa ferramenta permite ao criador do sistema preocupar-se apenas com a representação do conhecimento do perito. Encarrega-se também com a tarefa de interpretar o conhecimento representado e executá-lo em uma máquina, além de permitir depurações e explicações de como o computador chegou a uma ou mais conclusões preestabelecidas no sistema.

5.5 – Construção dos sistemas

5.5.1 – Sistema especialista para escolha da melhor dieta para um paciente cirúrgico

O trabalho de construção deste sistema foi facilitado na medida que os fatos e regras foram obtidos automaticamente a partir do fluxograma de decisão desenhado no SGR (Vide capítulos anteriores).

Para montagem do sistema especialista utilizar-se-á o "shell" Expert Sinta. O detalhamento de todas as etapas empregadas para criação do sistema, com a ferramenta, será mostrado a seguir.

O primeiro passo para a construção do sistema especialista é a obtenção de todas as variáveis e o conjunto de regras que compõem o sistema. Para este passo, como abordado nos capítulos anteriores, foi utilizado o 'Sistema Gerador de Regras'.

Com o relatório em mãos, gerado pelo SGR, inicia-se o Expert Sinta e após clique no botão 'Novo' ou a partir do menu a opção 'Arquivo' / 'Nova Base' um novo sistema especialista será criado (Figura 5. 1):

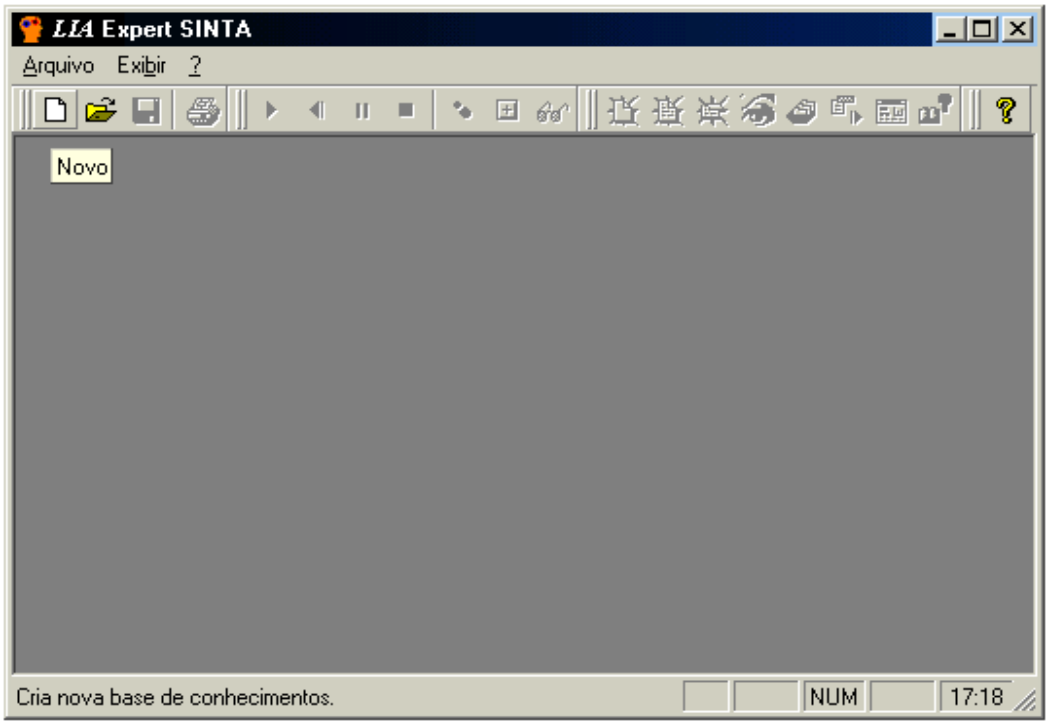


Figura 5. 1 – Tela de abertura do Expert Sinta com destaque para o botão 'Novo' que permite gerar uma nova base de conhecimentos.

Uma nova janela é exibida com as opções de manuseio das variáveis e das regras que comporão o sistema (Figura 5. 2).

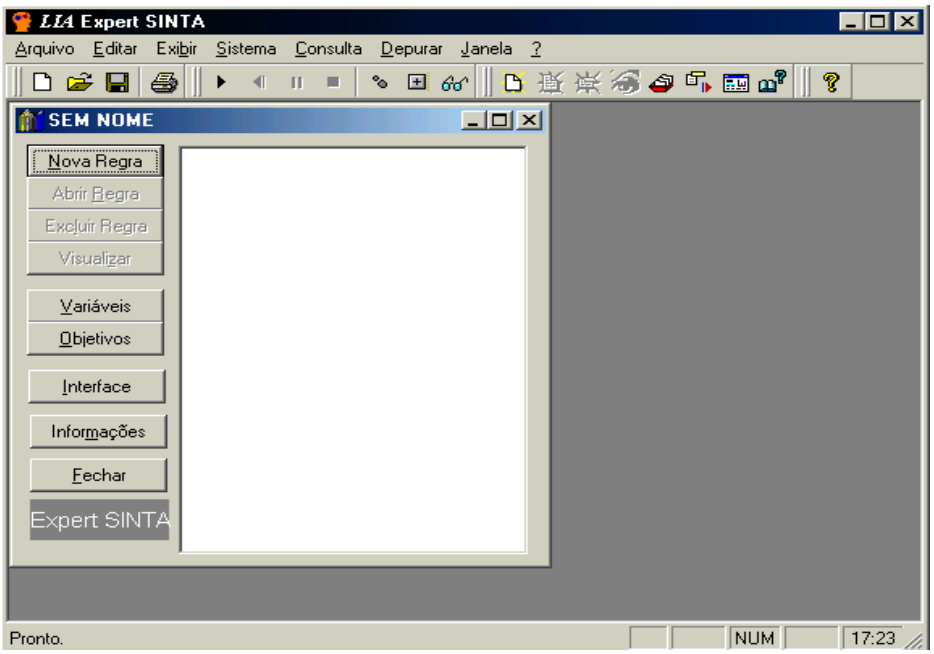


Figura 5. 2 – Janela para fornecimento dos componentes que constituirão o sistema especialista.

É obrigatório que, quando da geração de uma dada regra, todas as variáveis, que são componentes da cauda e da cabeça, sejam previamente fornecidas. Portanto, clica-se no botão 'Variáveis' e entra-se com o nome, tipo e valores que as variáveis poderão assumir ao longo de uma consulta (Figura 5. 3).

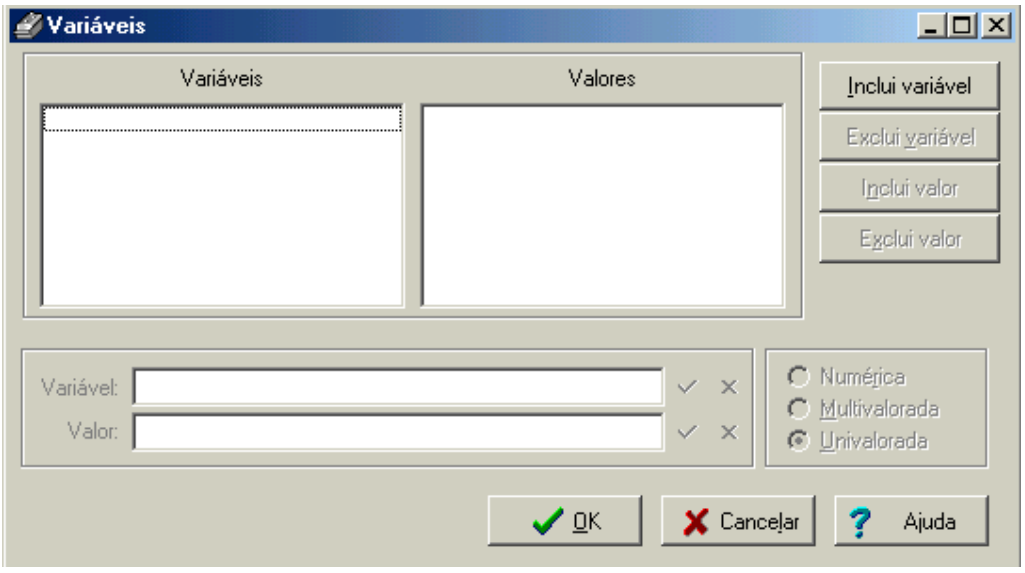


Figura 5. 3 – Janela de edição de variáveis do Expert Sinta.

Introduzem-se todas as variáveis, seus tipos e valores, nas caixas de textos correspondentes e à proporção que são feitas as admissões são necessárias as validações com clique na marca: '√' (Figura 5. 4).

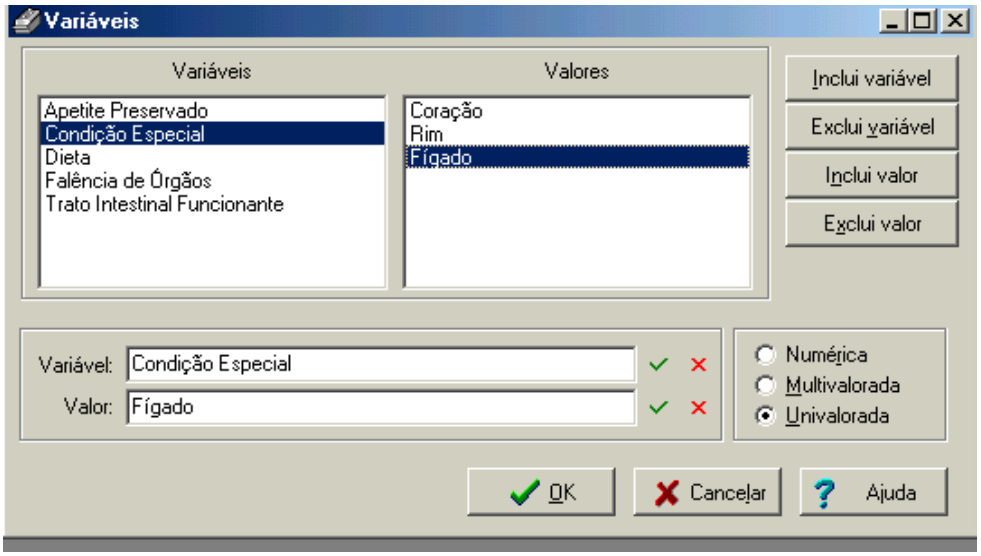


Figura 5. 4 – Variáveis do sistema especialista para escolha da melhor dieta para um paciente cirúrgico.

Quando todas as variáveis são incluídas clica-se no botão 'OK' e retorna-se para a tela da Figura 5. 2.

A seguir, são definidas as variáveis que serão objetivos do sistema, ou seja, aquelas que atuarão como peças conclusivas do sistema especialista. Para isto, clica-se no botão 'Objetivos' e na janela selecionam-se as variáveis, no lado direito, que são objetivos, para que as mesmas mudem de posição, passando para o quadro da direita, o que é obtido após clique nas setas de direção (Figura 5. 5).

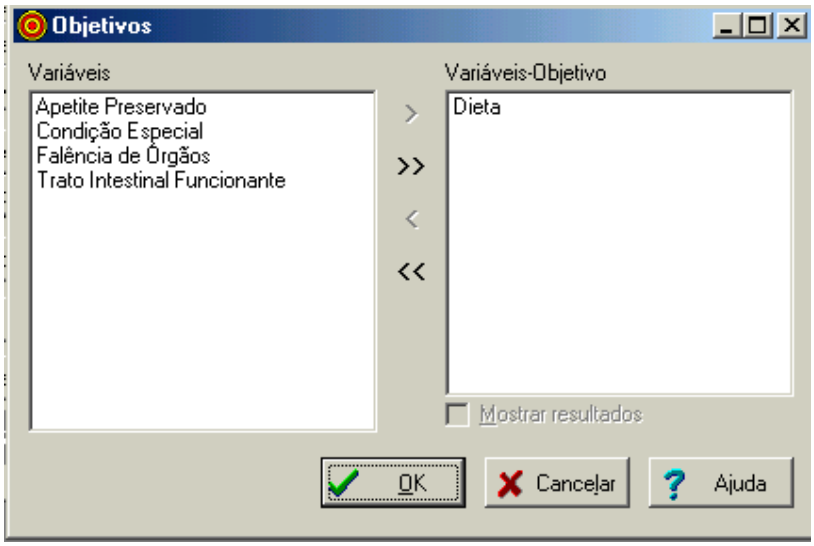


Figura 5. 5 – Definição das Variáveis-Objetivo.

Ao pressionar o botão 'OK' retorna-se para a janela da figura 5. 2. O passo seguinte é aquele que, definirá as perguntas que permitirão o encadeamento das regras do sistema especialista, para isto, clica-se no botão 'Interface' que resultará na exibição de nova janela (Figura 5. 6). Escolhem-se as variáveis que permitirão formulação de perguntas e transferirmo-las para o quadro: 'Variáveis com perguntas'; no quadro de edição apropriado preenche-se a caixa de edição com a pergunta relacionada aquela variável, conforme relatório gerado pelo SGR.

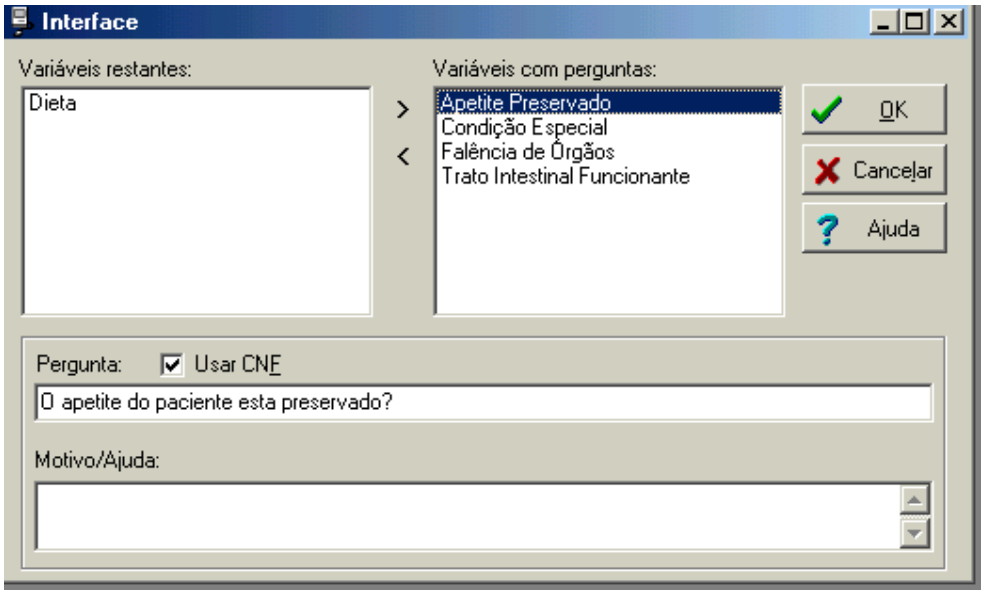


Figura 5. 6 – Edição de Perguntas / Motivos . Caso seja pretensão o uso de 'Crença' marca-se o quadro: Usar CNF.

Finalmente, após introdução das variáveis e suas propriedades (objetivo, perguntas, etc.) serão inseridas as regras que compõem o sistema.

Clica-se no Botão 'Nova Regra' que resultará na exibição da janela 'Nova Regra' (Figura 5. 7). Esta janela exibirá o número de ordem da regra e permite que regras anteriormente definidas possam servir de modelo, simplificando com isto, a elaboração de novas regras.

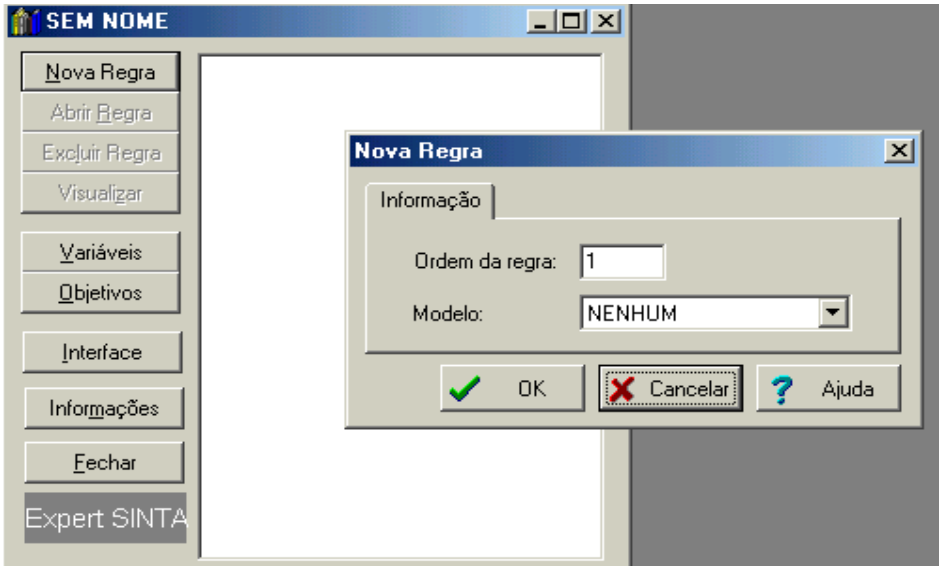


Figura 5. 7 – Nova Regra

Após definição, clica-se no botão 'OK' que permitirá a criação, edição ou modificação da regra (Figura 5. 8).

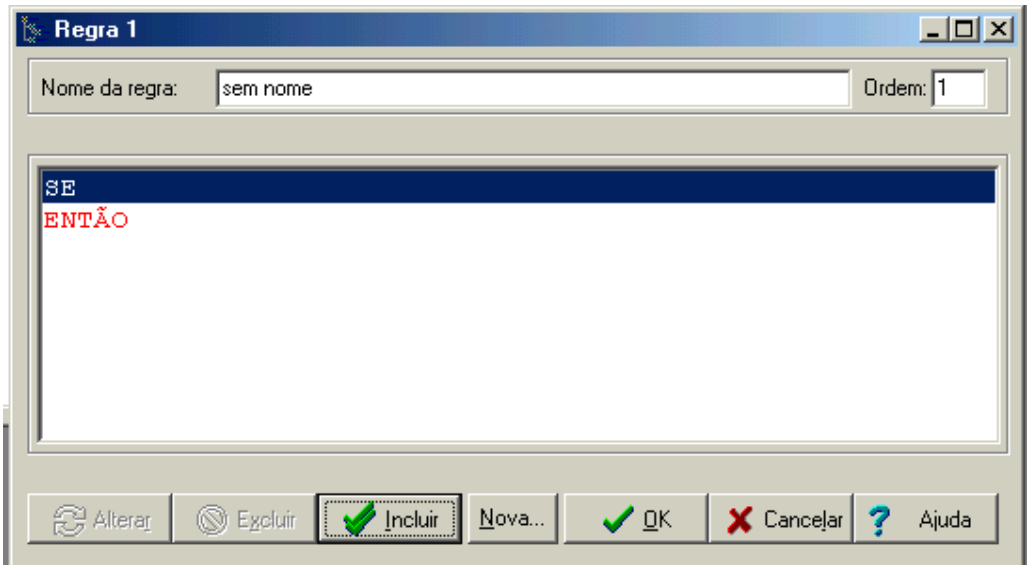


Figura 5. 8 – Janela para criação de regras

Nesta Janela é possível nomear a regra e alterar sua ordem. A alteração da numeração da regra terá influencia na avaliação desta regra pelo motor de inferência, quando no modo de consulta do sistema especialista.

Para introdução das premissas das regras, seleciona-se a opção 'SE' e clica-se no botão 'Incluir'. Nova janela será exibida (Figura 5. 9):

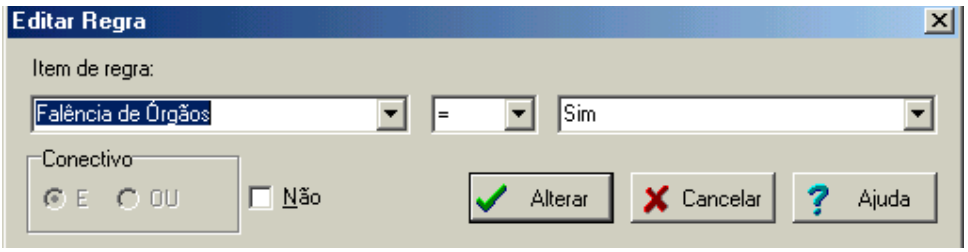


Figura 5. 9 – Janela de edição das premissas das regras

Como as variáveis e os seus valores foram previamente definidos introduzem-se os itens das regras a partir das caixas de seleção (Figura 5. 10). Os conectivos que permitirão a ligação das sentenças nas premissas também são definidos nesta etapa. Para cada sentença incluída na premissa clica-se o botão com a legenda 'Alterar'. As premissas são ligadas por conectivos. As regras obtidas a partir do SGR só admitem o conectivo "E".

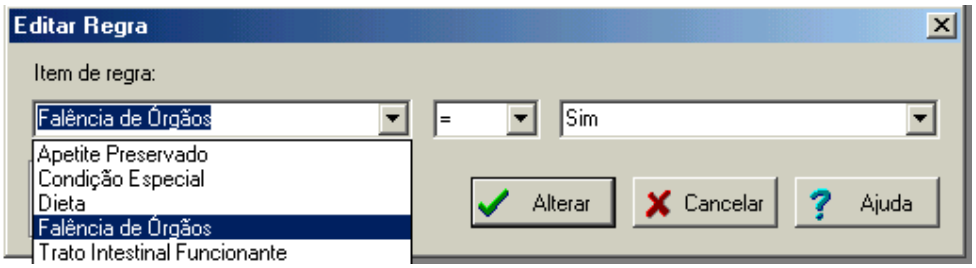


Figura 5. 10 – Uso das caixas de seleção

O procedimento para introdução da cabeça da regra é feito de modo semelhante, porém com seleção da opção 'ENTÃO' e clique no botão com legenda 'Inclui'. A janela para edição é a seguinte (Figura 5. 11):

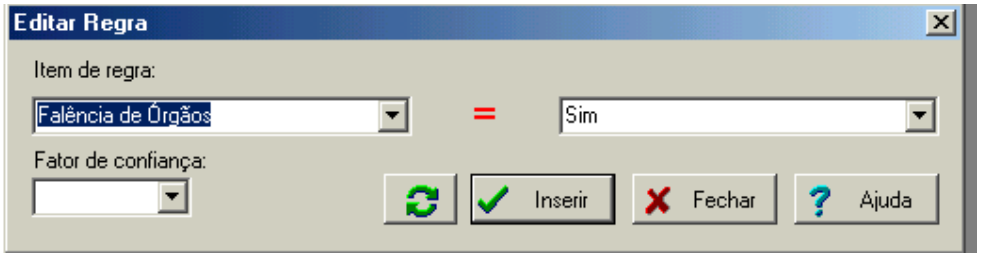


Figura 5. 11 – Edição da cabeça das regras

Caso seja disponível, introduz-se o 'Fator de Confiança' para que esta regra seja verdadeira.

Concluídas estas etapas, com a base de conhecimento completa, inicia-se o modo de consulta do Expert Sinta que permite a interação com o sistema. O sistema está incluído no CD-ROM anexo para que o leitor possa interagir quando desejar.

5.5.2 - Sistema especialista para diagnóstico de distúrbios hidroeletrolíticos

O sistema especialista para diagnóstico de distúrbios hidroeletrolíticos é dividido em dois módulos:

Diagnóstico Provável: o diagnóstico fornecido pelo programa é o mais provável para uma dada situação e é obtido a partir da interação com o usuário, que de modo passivo, informará os dados clínicos presentes em um determinado paciente, à proporção que perguntas são formuladas pelo sistema, no modo de consulta.

Diagnóstico Laboratorial: É um diagnóstico de certeza e é baseado no fornecimento dos valores laboratoriais dos eletrolíticos.

A escolha do módulo, desejado pelo usuário, é definido a partir da primeira pergunta, quando o sistema especialista é posto para funcionar em modo de consulta, e terá características de um menu de opção (Figura 5. 12). Para a criação do menu definiu-se uma variável univalorada (quando executamos o sistema teremos somente uma opção de consulta por vez: ou o diagnóstico provável ou o laboratorial) chamada 'Interface' com dois valores possíveis: 'Diagnóstico Provável' e 'Diagnóstico de Certeza'.

Na fase de construção do sistema, todas as regras que foram elaboradas para fornecimento de um diagnóstico de probabilidade obrigou à colocação nas premissas de uma sentença com a variável 'Interface' apresentando valor = Diagnóstico Provável. Nas premissas das regras elaboradas para conclusão por um Diagnóstico Laboratorial a variável 'Interface' assumiu valor = Diagnóstico de Certeza.

O Expert Sinta avalia as regras por ordem numérica, ou seja, a primeira a ser avaliada é a de número 1 se esta for refutada, a de número 2 é considerada e assim sucessivamente até que todas as regras sejam analisadas.

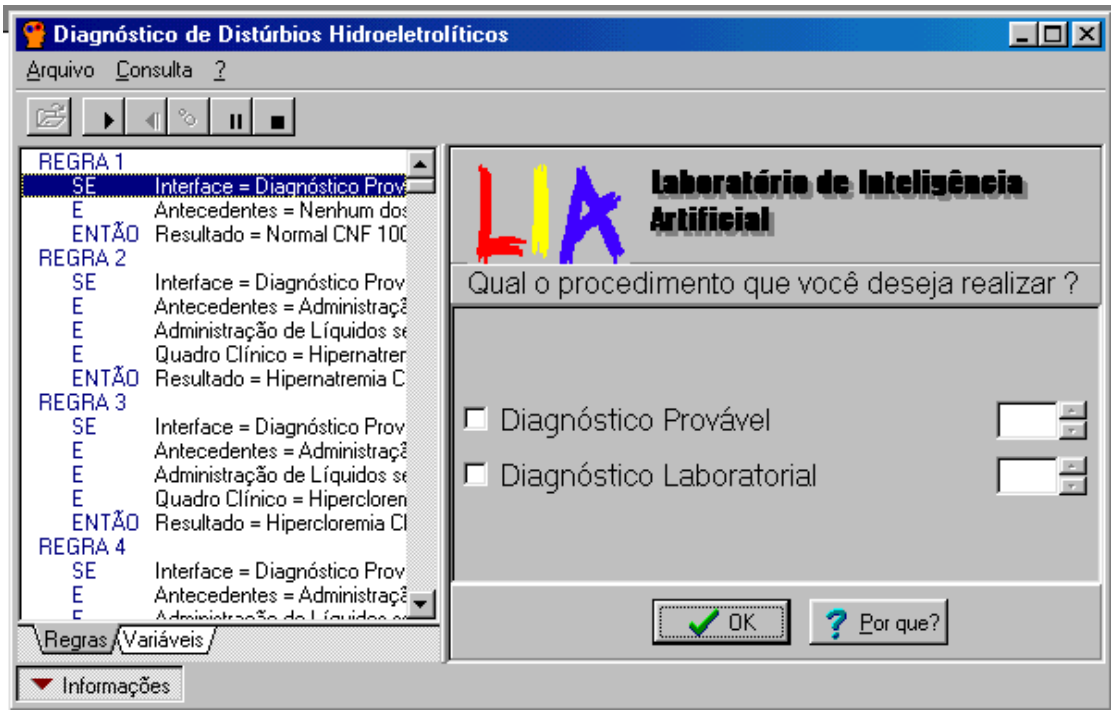


Figura 5. 12 – Tela inicial do sistema especialista para diagnóstico de distúrbios hidroeletrolíticos.

Quando o sistema especialista é posto para funcionar (modo de consulta), se a opção escolhida privilegie um 'Diagnóstico Provável' todas as regras que tiverem nas premissas a variável 'Interface' com valor = Diagnóstico de Certeza serão automaticamente descartadas e não mais serão chamadas para análise do quadro negro do sistema especialista. O sistema só levará em consideração, portanto, as regras elaboradas com a finalidade de obtenção de um 'Diagnóstico Provável'. Se a opção for pela marcação da caixa de seleção 'Diagnóstico Laboratorial' todas as regras que contenham nas premissas a variável Interface=Diagnóstico Provável serão descartadas e não serão chamadas para análise do quadro negro do sistema especialista. O sistema só levará em consideração, portanto, as regras elaboradas com a finalidade de obtenção de um 'Diagnóstico Laboratorial'.

5.5.2.1 - Módulo de Diagnóstico Provável:

Para que ocorra uma dada alteração hidroeletrolítica é necessária que haja um antecedente de perda ou ganho de água e/ou perda ou ganho de eletrólitos.

A perda ou ganho de um dado eletrólito não significa que o paciente irá apresentar uma situação clínica de excesso ou deficiência deste eletrolítico já que:

- É possível que o rim mantenha os níveis desse eletrólito em concentração sangüínea normalizada, seja fazendo eliminação, seja realizando retenção deste eletrólito.
- É possível que o paciente, mesmo perdendo um dado eletrólito, tenha o nível sangüíneo deste elemento elevado se simultaneamente estiver ocorrendo perda de água em nível proporcionalmente maior ao do eletrólito. Raciocínio semelhante pode ser empregado para paciente que, mesmo ganhando um dado eletrólito, tenha o seu valor plasmático diminuído devido ao recebimento simultâneo de doses aumentadas de água. Conclui-se que é possível ter aumento ou diminuição de níveis plasmáticos de um dado eletrólito de forma absoluta (ganho ou perda de eletrólitos isoladas) ou relativa (concentração e ou diluição plasmática).

Do exposto, pode-se observar que, para que um dado distúrbio hidroeletrolítico tenha tradução clínica, é necessário que além de antecedentes que levam aquela alteração exista a necessidade do paciente apresentar sintomatologia condizente com a alteração (tradução de que o organismo não foi capaz de compensar o distúrbio).

Observa-se ainda:

- Um mesmo achado clínico pode esta presente em mais de um distúrbio hidroeletrolítico, por exemplo, pode-se ter a sede como uma manifestação comum de desidratação, hipernatremia e hipercloremia.
- A ausência de um achado clínico teoricamente esperado para um dado distúrbio não afasta o diagnóstico (certos sinais e sintomas só se manifestam em uma fase mais avançado da alteração hidroeletrolítica).
- A presença de um determinado achado clínico pode ser observada tanto em situações que cursam com aumento como também em diminuição do valor plasmático de um dado eletrólito, como conseqüência, a elaboração de regras que levassem em consideração um antecedente e um achado clínico desse tipo levaria a conclusões inconsistentes como, por exemplo, um paciente apresentar hipo e hiperpotasemia numa mesma consulta (aumento e diminuição de potássio plasmático ao mesmo tempo).

As inconsistências são diminuídas admiravelmente se, no processo de criação das regras, ao invés de levar-se em consideração achados clínicos isolados levando a um determinado distúrbio, adotar-se:

- Criação de uma variável chamada 'Quadro Clínico' com valores correspondentes as possíveis alterações do equilíbrio hidroeletrolítico.
- As premissas são formadas pelo conjunto de sinais e sintomas que levam a concluir por uma determinada alteração hidroeletrolítica.
- Regras são elaboradas contendo em suas premissas o conjunto dos achados clínicos e como conclusão a variável multivalorada 'Quadro Clínico' com valor correspondente ao distúrbio hidroeletrolítico resultante.
- A variável 'Quadro Clínico' não é do tipo objetivo, portanto, não será exibida como um estado meta no modo de consulta do sistema especialista.

Era desejo que as regras apresentassem crenças baseadas em dados estatísticos confiáveis, no entanto, não foi possível obtenção, na literatura, de ocorrências do tipo:

Se sede então Diagnóstico = desidratação CNF = 90%.

Devido a não obrigatoriedade da presença de todos os achados clínicos para caracterizar um dado distúrbio e, como relatado anteriormente, a ausência na literatura do percentual de presença ou ausência de cada um dos sinais e sintomas, optou-se pelo seguinte tratamento na formação das premissas na construção das regras:

- Conectivo ‘e’: antecedendo as variáveis cujos valores estão presentes em quase 100% das situações.
- Conectivo ‘ou’: Antecedendo achados clínicos eventualmente encontrados ou presentes em situações mais graves da alteração hidroeletrolítica, portanto em um percentual bem menor de casos.

Os conjuntos das regras para este módulo são:

- Regras de Classificação:

Regra 1
SE Interface = Diagnóstico Provável
E Antecedentes = Nenhum dos Anteriores
ENTÃO Resultado = Normal CNF 100%

Regra 2
SE Interface = Diagnóstico Provável
E Antecedentes = Administração de líquidos sem sais
E Administração de Líquidos sem sais = Deficiente (Preso, naufrago. coma, disfagia total (Megaesôfago, Câncer de esôfago))
E Quadro Clínico = Hipernatremia
ENTÃO Resultado = Hipernatremia CNF 100%

Regra 3
SE Interface = Diagnóstico Provável
E Antecedentes = Administração de líquidos sem sais
E Administração de Líquidos sem sais = Deficiente (Preso, naufrago. coma, disfagia total (Megaesôfago, Câncer de esôfago))
E Quadro Clínico = Hipercloremia
ENTÃO Resultado = Hipercloremia CNF 100%

Regra 4
SE Interface = Diagnóstico Provável
E Antecedentes = Administração de líquidos sem sais
E Administração de Líquidos sem sais = Deficiente (Preso, naufrago. coma, disfagia total (Megaesôfago, Câncer de esôfago))
E Quadro Clínico = Hiperpotassemia
ENTÃO Resultado = Hiperpotassemia CNF 100%

Regra 5
SE Interface = Diagnóstico Provável
E Antecedentes = Administração de líquidos sem sais
E Administração de Líquidos sem sais = Deficiente (Preso, naufrago. coma, disfagia total (Megaesôfago, Câncer de esôfago))
E Quadro Clínico = Desidratação
ENTÃO Resultado = Desidratação CNF 100%

Regra 6
SE Interface = Diagnóstico Provável
E Antecedentes = Administração de líquidos sem sais
E Administração de Líquidos sem sais = Excessiva
E Quadro Clínico = Hiponatremia
ENTÃO Resultado = Hiponatremia CNF 100%

Regra 7
SE Interface = Diagnóstico Provável
E Antecedentes = Administração de líquidos sem sais
E Administração de Líquidos sem sais = Excessiva
E Quadro Clínico = Hipocloremia
ENTÃO Resultado = Hipocloremia CNF 100%

Regra 8
SE Interface = Diagnóstico Provável
E Antecedentes = Administração de líquidos sem sais
E Administração de Líquidos sem sais = Excessiva
E Quadro Clínico = Hiperhidratação celular

ENTÃO Resultado = Hiperhidratação celular CNF 100%

Regra 9
SE Interface = Diagnóstico Provável
E Antecedentes = Administração de sais
E Administração de sais = Excessiva
E Administração Excessiva de Sais = Sódio
E Quadro Clínico = Hipernatremia
ENTÃO Resultado = Hipernatremia CNF 100%

Regra 10
SE Interface = Diagnóstico Provável
E Antecedentes = Administração de sais
E Administração de sais = Excessiva
E Administração Excessiva de Sais = Cloro
E Quadro Clínico = Hipercloremia
ENTÃO Resultado = Hipercloremia CNF 100%

Regra 11
SE Interface = Diagnóstico Provável
E Antecedentes = Administração de sais
E Administração de sais = Excessiva
E Administração Excessiva de Sais = Potássio
E Quadro Clínico = Hiperpotassemia
ENTÃO Resultado = Hiperpotassemia CNF 100%

Regra 12
SE Interface = Diagnóstico Provável
E Antecedentes = Administração de sais
E Administração de sais = Deficiente
E Deficiência de sal = Cloro
E Quadro Clínico = Hipocloremia
ENTÃO Resultado = Hipocloremia CNF 100%

Regra 13
SE Interface = Diagnóstico Provável
E Antecedentes = Administração de sais
E Administração de sais = Deficiente
E Deficiência de sal = Sódio
E Quadro Clínico = Hiponatremia
ENTÃO Resultado = Hiponatremia CNF 100%

Regra 14
SE Interface = Diagnóstico Provável
E Antecedentes = Administração de sais
E Administração de sais = Deficiente
E Deficiência de sal = Potássio
E Quadro Clínico = Hipopotassemia
ENTÃO Resultado = Hipopotassemia CNF 100%

Regra 15
SE Interface = Diagnóstico Provável
E Antecedentes = Alterações do trato Gastro-Intestinal
E Alteração do Trato Gastro-Intestinal = Perdas Gástricas (Vômito - Aspiração)
E Quadro Clínico = Hiponatremia
ENTÃO Resultado = Hiponatremia CNF 100%

Regra 16
SE Interface = Diagnóstico Provável
E Antecedentes = Alterações do trato Gastro-Intestinal
E Alteração do Trato Gastro-Intestinal = Perdas Gástricas (Vômito - Aspiração)
E Quadro Clínico = Hipocloremia
ENTÃO Resultado = Hipocloremia CNF 100%

Regra 17
SE Interface = Diagnóstico Provável
E Antecedentes = Alterações do trato Gastro-Intestinal
E Alteração do Trato Gastro-Intestinal = Perdas Gástricas (Vômito - Aspiração)
E Quadro Clínico = Hiponatremia
ENTÃO Resultado = Hiponatremia CNF 100%

Regra 18

SE Interface = Diagnóstico Provável
E Antecedentes = Alterações do trato Gastro-Intestinal
E Alteração do Trato Gastro-Intestinal = Perdas Gástricas (Vômito - Aspiração)
E Quadro Clínico = Desidratação
ENTÃO Resultado = Desidratação CNF 100%

Regra 19
SE Interface = Diagnóstico Provável
E Antecedentes = Alterações do trato Gastro-Intestinal
E Alteração do Trato Gastro-Intestinal = Lavagem intestinal, diarréia e/ou Íleo
E Quadro Clínico = Hiponatremia
ENTÃO Resultado = Hiponatremia CNF 100%

Regra 20
SE Interface = Diagnóstico Provável
E Antecedentes = Alterações do trato Gastro-Intestinal
E Alteração do Trato Gastro-Intestinal = Lavagem intestinal, diarréia e/ou Íleo
E Quadro Clínico = Hipocloremia
ENTÃO Resultado = Hipocloremia CNF 100%

Regra 21
SE Interface = Diagnóstico Provável
E Antecedentes = Alterações do trato Gastro-Intestinal
E Alteração do Trato Gastro-Intestinal = Lavagem intestinal, diarréia e/ou Íleo
E Quadro Clínico = Hipopotassemia
ENTÃO Resultado = Hipopotassemia CNF 100%

Regra 22
SE Interface = Diagnóstico Provável
E Antecedentes = Alterações do trato Gastro-Intestinal
E Alteração do Trato Gastro-Intestinal = Lavagem intestinal, diarréia e/ou Íleo
E Quadro Clínico = Desidratação
ENTÃO Resultado = Desidratação CNF 100%

Regra 23
SE Interface = Diagnóstico Provável
E Antecedentes = Alterações do trato Gastro-Intestinal
E Alteração do Trato Gastro-Intestinal = Fístulas - Gástrica, Pancreática, Biliar, Intestinal.
E Quadro Clínico = Hiponatremia
ENTÃO Resultado = Hiponatremia CNF 100%

Regra 24
SE Interface = Diagnóstico Provável
E Antecedentes = Alterações do trato Gastro-Intestinal
E Alteração do Trato Gastro-Intestinal = Fístulas - Gástrica, Pancreática, Biliar, Intestinal.
E Quadro Clínico = Hipocloremia
ENTÃO Resultado = Hipocloremia CNF 100%

Regra 25
SE Interface = Diagnóstico Provável
E Antecedentes = Alterações do trato Gastro-Intestinal
E Alteração do Trato Gastro-Intestinal = Fístulas - Gástrica, Pancreática, Biliar, Intestinal.
E Quadro Clínico = Hipopotassemia
ENTÃO Resultado = Hipopotassemia CNF 100%

Regra 26
SE Interface = Diagnóstico Provável
E Antecedentes = Alterações do trato Gastro-Intestinal
E Alteração do Trato Gastro-Intestinal = Fístulas - Gástrica, Pancreática, Biliar, Intestinal.
E Quadro Clínico = Desidratação
ENTÃO Resultado = Desidratação CNF 100%

Regra 27
SE Interface = Diagnóstico Provável
E Antecedentes = Alterações do trato Gastro-Intestinal
E Alteração do Trato Gastro-Intestinal = Anastomose Uretero Intestinal bilateral.
E Quadro Clínico = Hipercloremia
ENTÃO Resultado = Hipercloremia CNF 100%

Regra 28
SE Interface = Diagnóstico Provável
E Antecedentes = Perdas Urinárias (Poliúria, Uso de Diuréticos)

E Quadro Clínico = Hiponatremia
ENTÃO Resultado = Hiponatremia CNF 100%

Regra 29
SE Interface = Diagnóstico Provável
E Antecedentes = Perdas Urinárias (Poliúria, Uso de Diuréticos)
E Quadro Clínico = Hipocloremia
ENTÃO Resultado = Hipocloremia CNF 100%

Regra 30
SE Interface = Diagnóstico Provável
E Antecedentes = Perdas Urinárias (Poliúria, Uso de Diuréticos)
E Quadro Clínico = Hipopotassemia
ENTÃO Resultado = Hipopotassemia CNF 100%

Regra 31
SE Interface = Diagnóstico Provável
E Antecedentes = Perdas Urinárias (Poliúria, Uso de Diuréticos)
E Quadro Clínico = Desidratação
ENTÃO Resultado = Desidratação CNF 100%

Regra 32
SE Interface = Diagnóstico Provável
E Antecedentes = Perdas através da Pele e dos Pulmões
E Perdas Através da pele e pulmão = Sudorese intensa
E Quadro Clínico = Hipocloremia
ENTÃO Resultado = Hipocloremia CNF 100%

Regra 33
SE Interface = Diagnóstico Provável
E Antecedentes = Perdas através da Pele e dos Pulmões
E Perdas Através da pele e pulmão = Sudorese intensa
E Quadro Clínico = Hiponatremia
ENTÃO Resultado = Hiponatremia CNF 100%

Regra 34
SE Interface = Diagnóstico Provável
E Antecedentes = Perdas através da Pele e dos Pulmões
E Perdas Através da pele e pulmão = Sudorese intensa
E Quadro Clínico = Desidratação
ENTÃO Resultado = Desidratação CNF 100%

Regra 35
SE Interface = Diagnóstico Provável
E Antecedentes = Perdas através da Pele e dos Pulmões
E Perdas Através da pele e pulmão = Taquipnéia / Asma
E Quadro Clínico = Desidratação
ENTÃO Resultado = Desidratação CNF 100%

Regra 36
SE Interface = Diagnóstico Provável
E Antecedentes = Perdas através da Pele e dos Pulmões
E Perdas Através da pele e pulmão = Queimaduras
E Quadro Clínico = Hiponatremia
ENTÃO Resultado = Hiponatremia CNF 100%

Regra 37
SE Interface = Diagnóstico Provável
E Antecedentes = Perdas através da Pele e dos Pulmões
E Perdas Através da pele e pulmão = Queimaduras
E Quadro Clínico = Hipocloremia
ENTÃO Resultado = Hipocloremia CNF 100%

Regra 38
SE Interface = Diagnóstico Provável
E Antecedentes = Perdas através da Pele e dos Pulmões
E Perdas Através da pele e pulmão = Queimaduras
E Quadro Clínico = Hiperpotassemia
ENTÃO Resultado = Hiperpotassemia CNF 100%

Regra 39
SE Interface = Diagnóstico Provável

E Antecedentes = Perdas através da Pele e dos Pulmões
E Perdas Através da pele e pulmão = Queimaduras
E Quadro Clínico = Desidratação
ENTÃO Resultado = Desidratação CNF 100%

Regra 40
SE Interface = Diagnóstico Provável
E Antecedentes = Doenças Hormonais e/ou Metabólicas
E Alteração Hormonal e/ou metabólica = Uso de Insulina + Glicose
E Quadro Clínico = Hipopotassemia
ENTÃO Resultado = Hipopotassemia CNF 100%

Regra 41
SE Interface = Diagnóstico Provável
E Antecedentes = Doenças Hormonais e/ou Metabólicas
E Alteração Hormonal e/ou metabólica = Distúrbios do Hormônio Anti-Diurético (Diabetes incididos)
E Quadro Clínico = Hipernatremia
ENTÃO Resultado = Hipernatremia CNF 100%

Regra 42
SE Interface = Diagnóstico Provável
E Antecedentes = Doenças Hormonais e/ou Metabólicas
E Alteração Hormonal e/ou metabólica = Distúrbios do Hormônio Anti-Diurético (Diabetes incididos)
E Quadro Clínico = Desidratação
ENTÃO Resultado = Desidratação CNF 100%

Regra 43
SE Interface = Diagnóstico Provável
E Antecedentes = Doenças Hormonais e/ou Metabólicas
E Alteração Hormonal e/ou metabólica = Hiperaldosteronismo Primário
E Quadro Clínico = Hipernatremia
ENTÃO Resultado = Hipernatremia CNF 100%

Regra 44
SE Interface = Diagnóstico Provável
E Antecedentes = Doenças Hormonais e/ou Metabólicas
E Alteração Hormonal e/ou metabólica = Doença de Addison
E Quadro Clínico = Hiperpotassemia
ENTÃO Resultado = Hiperpotassemia CNF 100%

Regra 45
SE Interface = Diagnóstico Provável
E Antecedentes = Doenças Hormonais e/ou Metabólicas
E Alteração Hormonal e/ou metabólica = Cetoacidose diabética, Mucovicidose, Insuficiência Supra-Renal
E Quadro Clínico = Desidratação
ENTÃO Resultado = Desidratação CNF 100%

Regra 46
SE Interface = Diagnóstico Provável
E Antecedentes = Doenças Hormonais e/ou Metabólicas
E Alteração Hormonal e/ou metabólica = Hipoproteinemia
E Quadro Clínico = Hiperhidratação extracelular
ENTÃO Resultado = Hiperhidratação extracelular CNF 100%

Regra 47
SE Interface = Diagnóstico Provável
E Antecedentes = Doenças Hormonais e/ou Metabólicas
E Alteração Hormonal e/ou metabólica = Hipoproteinemia
E Quadro Clínico = Hiponatremia
ENTÃO Resultado = Hiponatremia CNF 100%

Regra 48
SE Interface = Diagnóstico Provável
E Antecedentes = Doenças Hormonais e/ou Metabólicas
E Alteração Hormonal e/ou metabólica = Hipoproteinemia
E Quadro Clínico = Hipocloremia
ENTÃO Resultado = Hipocloremia CNF 100%

Regra 49
SE Interface = Diagnóstico Provável
E Antecedentes = Insuficiência Cardíaca
E Quadro Clínico = Hiperhidratação extracelular

ENTÃO Resultado = Hiperhidratação extracelular CNF 100%

Regra 50
SE Interface = Diagnóstico Provável
E Antecedentes = Insuficiência Cardíaca
E Quadro Clínico = Hiponatremia
ENTÃO Resultado = Hiponatremia CNF 100%

Regra 51
SE Interface = Diagnóstico Provável
E Antecedentes = Insuficiência Cardíaca
E Quadro Clínico = Hipocloremia
ENTÃO Resultado = Hipocloremia CNF 100%

Regra 52
SE Interface = Diagnóstico Provável
E Antecedentes = Insuficiência Renal
E Quadro Clínico = Hipercloremia
ENTÃO Resultado = Hipercloremia CNF 100%

Regra 53
SE Interface = Diagnóstico Provável
E Antecedentes = Insuficiência Renal
E Quadro Clínico = Hipernatremia
ENTÃO Resultado = Hipernatremia CNF 100%

Regra 54
SE Interface = Diagnóstico Provável
E Antecedentes = Insuficiência Renal
E Quadro Clínico = Hiperpotassemia
ENTÃO Resultado = Hiperpotassemia CNF 100%

Regra 55
SE Interface = Diagnóstico Provável
E Antecedentes = Insuficiência Renal
E Quadro Clínico = Hiperhidratação extracelular
ENTÃO Resultado = Hiperhidratação extracelular CNF 100%

Regra 56
SE Interface = Diagnóstico Provável
E Antecedentes = Hemorragias
E Quadro Clínico = Desidratação
ENTÃO Resultado = Desidratação CNF 100%

Regra 57
SE Interface = Diagnóstico Provável
E Antecedentes = Alterações do Metabolismo Ácido-Básico
E Estado metabólico = Acidose
E Acidose = Metabólica
E Quadro Clínico = Hiperpotassemia
ENTÃO Resultado = Hiperpotassemia CNF 100%

Regra 58
SE Interface = Diagnóstico Provável
E Antecedentes = Alterações do Metabolismo Ácido-Básico
E Estado metabólico = Alcalose
E Alcalose = Metabólica
E Quadro Clínico = Hipopotassemia
ENTÃO Resultado = Hipopotassemia CNF 100%

Regra 59
SE Interface = Diagnóstico Provável
E Antecedentes = Alterações do Metabolismo Ácido-Básico
E Estado metabólico = Acidose
E Acidose = Respiratória
E Quadro Clínico = Hipocloremia
ENTÃO Resultado = Hipocloremia CNF 100%

Regra 60
SE Interface = Diagnóstico Provável
E Antecedentes = Alterações do Metabolismo Ácido-Básico
E Estado metabólico = Alcalose

E Alcalose = Respiratória
E Quadro Clínico = Hipopotassemia
ENTÃO Resultado = Hipopotassemia CNF 100%

Regra 61
SE Interface = Diagnóstico Provável
E Antecedentes = Alterações do Metabolismo Ácido-Básico
E Estado metabólico = Acidose
E Acidose = Respiratória
E Quadro Clínico = Hiperpotassemia
ENTÃO Resultado = Hiperpotassemia CNF 100%

Regra 62
SE Interface = Diagnóstico Provável
E Antecedentes = Alterações do Metabolismo Ácido-Básico
E Estado metabólico = Alcalose
E Alcalose = Respiratória
E Quadro Clínico = Hipercloremia
ENTÃO Resultado = Hipercloremia CNF 100%

- Regras cuja cabeça não é objetivo do sistema

Regra 63
SE Interface = Diagnóstico Provável
E Sede = Sim
OU Músculo = Espasmos musculares amplos
OU Febre = Sim
OU Confusão Mental e Torpor = Sim
OU Lassidão = Sim
OU Coma = Sem Agitação
ENTÃO Quadro Clínico = Hipercloremia CNF 100%

Regra 64
SE Interface = Diagnóstico Provável
E Músculo = Fraqueza
OU Sensibilidade = parestesia
OU Sensibilidade = Anestesia
OU Freqüência cardíaca = Bradicardia
OU Arritmia = Fibrilação Ventricular
OU Arritmia = Parada cardíaca em diástole
OU ECG = Complexo QRS alargado
OU ECG = Onda T apiculada e simétrica
OU Excitação = Sim
OU Ansiedade = Sim
OU Agitação = Sim
OU Torpor = Sim
ENTÃO Quadro Clínico = Hiperpotassemia CNF 100%

Regra 65
SE Interface = Diagnóstico Provável
E Sede = Sim
E Diurese = Diminuída
OU Freqüência cardíaca = Taquicardia
OU Turgor e elasticidade da pele = Sim
OU Arritmia = Fibrilação Ventricular
OU Pressão Arterial = Hipotensão arterial
OU Vômitos = Sim
OU Músculo = Fraqueza
OU Fontanelas deprimidas = Sim
OU Olhos encovados = Sim
OU Variação de Peso = Diminuição
OU Pulso = Fino
OU Pressão venosa = Diminuída (Veias colabadas)
OU Apatia = Sim
OU Tonturas = Sim
OU Tonturas = Sim
OU Irritabilidade = Sim
ENTÃO Quadro Clínico = Hiperpotassemia CNF 100%

Regra 66
SE Interface = Diagnóstico Provável
E Sede = Sim

OU Diurese = Diminuída
OU Mucosas = Secas
OU Febre = Sim
OU Confusão Mental e Torpor = Sim
OU Delírio = Sim
OU Coma = Com agitação
ENTÃO Quadro Clínico = Hipernatremia CNF 100%

Regra 67
SE Interface = Diagnóstico Provável
E Freqüência cardíaca = Taquicardia
OU Pressão Arterial = Hipotensão arterial
OU Músculo = Fraqueza
OU Pulso = Fino
OU Extremidades Frias = Sim
OU Anorexia = Sim
OU Náuseas = Sim
OU Vômitos = Sim
OU Apatia = Sim
OU Lassidão = Sim
OU Confusão Mental e Torpor = Sim
OU Coma = Sem Agitação
OU Cefaléia = Sim
ENTÃO Quadro Clínico = Hiponatremia CNF 100%

Regra 68
SE Interface = Diagnóstico Provável
OU Constipação, distensão abdominal, íleo = Sim
OU Vômitos = Sim
OU Pressão Arterial = Hipotensão arterial
OU Pressão venosa = Diminuída (Veias colabadas)
ENTÃO Quadro Clínico = Hipocloremia CNF 100%

Regra 69
SE Interface = Diagnóstico Provável
E Constipação, distensão abdominal, íleo = Sim
OU Músculo = Fraqueza
OU Músculo = Hipotonia
OU Sensibilidade = paralisia
OU Pressão Arterial = Hipertensão arterial
OU Arritmia = Extrassistoles
OU Arritmia = Parada cardíaca em diástole
OU ECG = Depressão ST
OU ECG = Onda U elevada
OU ECG = QRS alto
OU ECG = Onda T reduzida, aspecto difásico ou invertida
OU Apatia = Sim
OU Delírio = Sim
OU Coma = Sem Agitação
ENTÃO Quadro Clínico = Hipopotassemia CNF 100%

Regra 70
SE Interface = Diagnóstico Provável
E Hipertensão Intracraniana (Cefaléia, Vômitos, Hipertensão Arterial, Sinal de Babinski) = Sim
OU Aspecto do Paciente = Edemaciado
OU Ausculta Pulmonar = Estertores
OU Derrames cavitários (Ascite, Derrame Pleural) = Sim
OU Variação de Peso = Aumento
ENTÃO Quadro Clínico = Hiperhidratação celular CNF 100%

Regra 71
SE Interface = Diagnóstico Provável
E Pressão Arterial = Hipertensão arterial
E Pressão venosa = Aumentada (Jugular Túrgida)
OU Variação de Peso = Aumento
OU Derrames cavitários (Ascite, Derrame Pleural) = Sim
OU Ausculta Pulmonar = Estertores
OU Ausculta Pulmonar = Dispnéia
OU Mucosas = Cianose
OU Arritmia = Ritmo de Galope
ENTÃO Quadro Clínico = Hiperhidratação extracelular CNF 100%

5.5.2.2 - Módulo de Diagnóstico Laboratorial

Para este módulo, o trabalho é simplificado já que é possível a geração de regras a partir dos fluxogramas de decisão desenhados no SGR (Figuras 5. 13, 5. 14 e 5. 15)

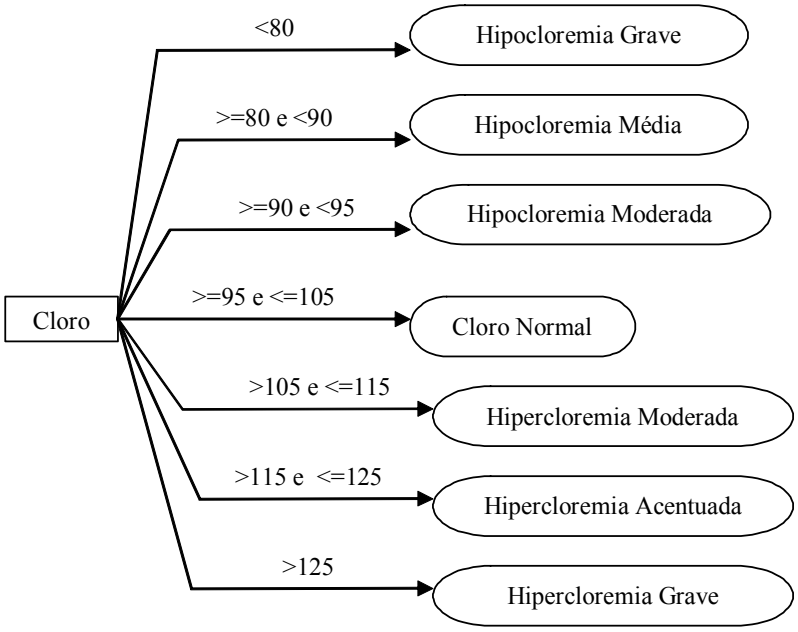


Figura 5. 13 – Diagnóstico laboratorial de alterações plasmáticas do cloro.

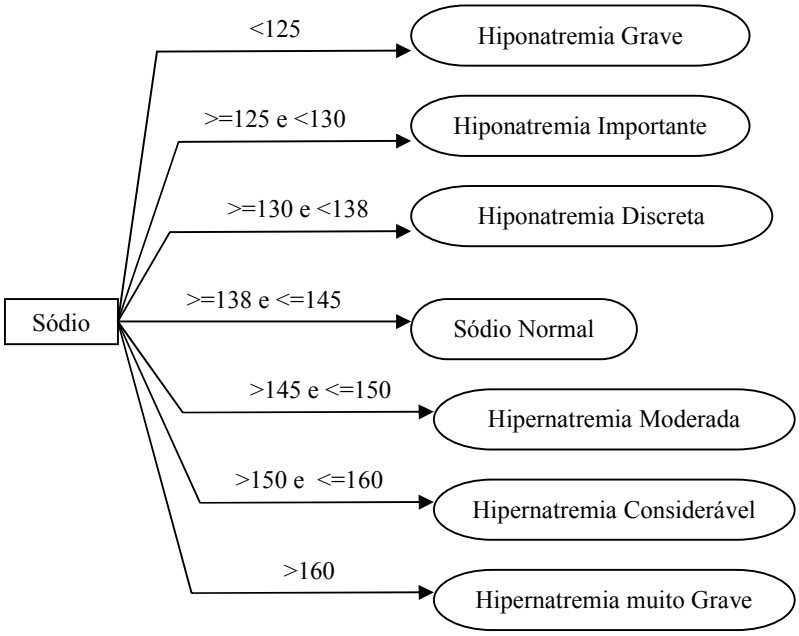


Figura 5. 14 - Diagnóstico laboratorial de alterações plasmáticas do sódio.

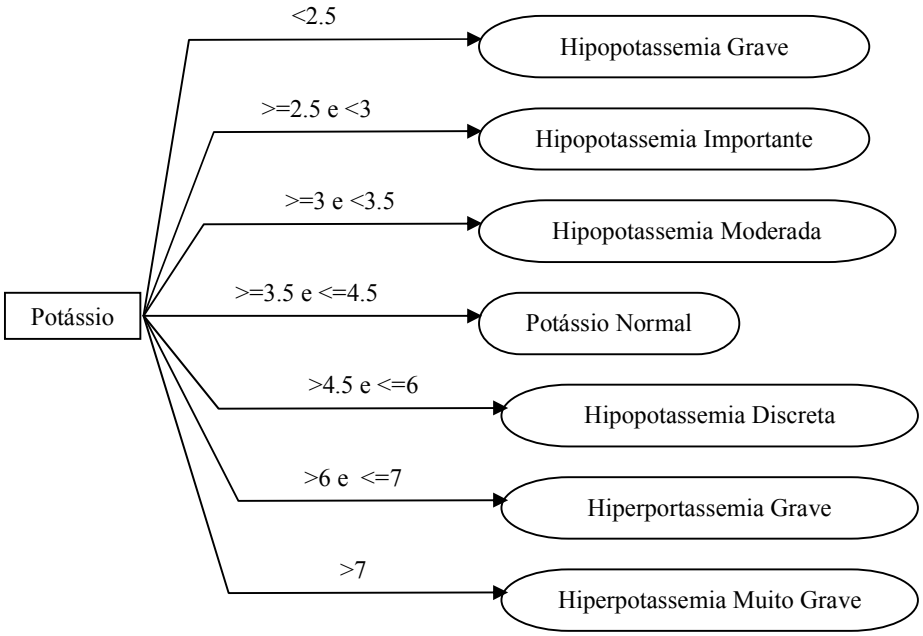


Figura 5. 15 - Diagnóstico laboratorial de alterações plasmáticas do potássio.

Quando desenhado no SGR motivou a exibição da seguinte tela (Figura 5. 16):

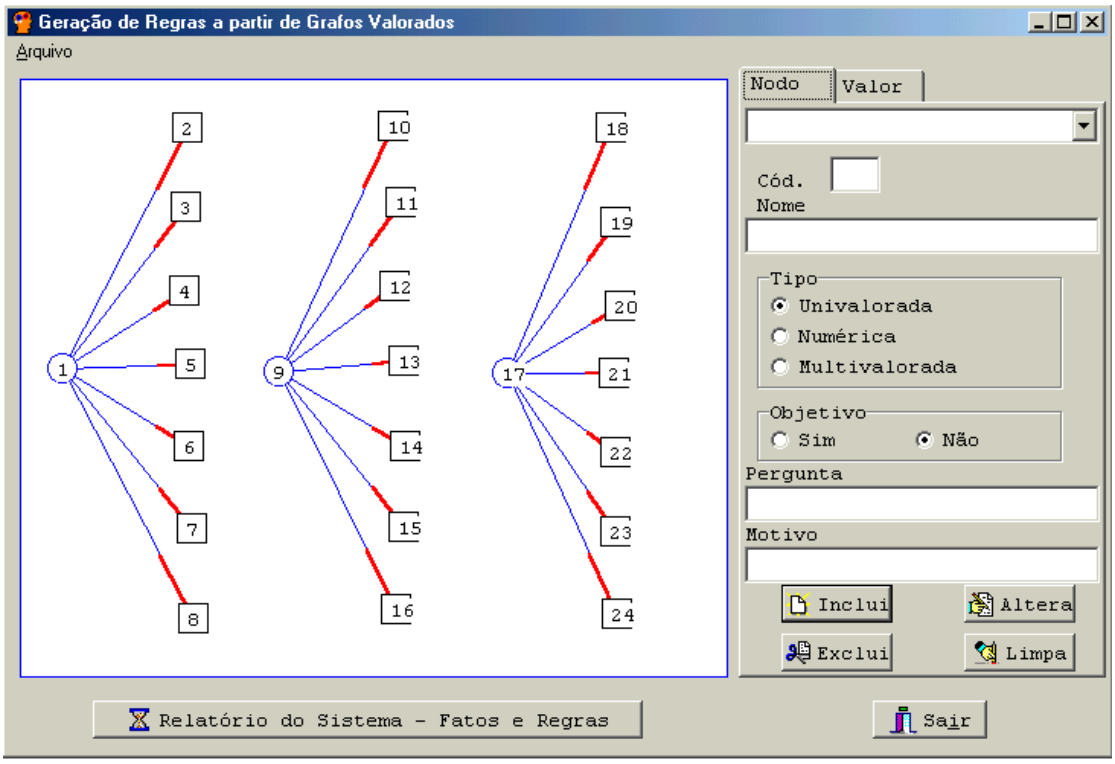


Figura 5. 16 – Desenho dos fluxogramas para obtenção do diagnóstico laboratorial no SGR

Que gerou as regras:

```
REGRA 1
  SE
       Cloro < 80
          ENTÃO
          Diagnóstico = Hipocloremia Grave

REGRA 2
  SE
       Cloro >= 80     < 90
          ENTÃO
          Diagnóstico = Hipocloremia Média

REGRA 3
  SE
       Cloro >= 90     < 95
          ENTÃO
          Diagnóstico = Hipocloremia Moderada

REGRA 4
  SE
       Cloro >= 95     <= 105
          ENTÃO
          Diagnóstico = Cloro Normal

REGRA 5
  SE
       Cloro > 105     <= 115
          ENTÃO
          Diagnóstico = Hipercloremia Moderada

REGRA 6
  SE
       Cloro > 115     <= 125
          ENTÃO
          Diagnóstico = Hipercloremia Acentuada

REGRA 7
  SE
       Cloro > 125
          ENTÃO
          Diagnóstico = Hipercloremia Grave

REGRA 8
  SE
       Sódio < 125
          ENTÃO
          Diagnóstico = Hiponatremia Grave

REGRA 9
  SE
       Sódio >= 125     < 130
          ENTÃO
          Diagnóstico = Hiponatremia Importante

REGRA 10
  SE
       Sódio >= 130     <= 138
          ENTÃO
          Diagnóstico = Hiponatremia Discreta

REGRA 11
  SE
       Sódio >= 138     <= 145
          ENTÃO
          Diagnóstico = Sódio Normal

REGRA 12
  SE
       Sódio > 145     <= 150
          ENTÃO
          Diagnóstico = Hipernatremia Moderada
```

```
REGRA 13
  SE
       Sódio > 150     <= 160
          ENTÃO
          Diagnóstico = Hipernatremia Considerável

REGRA 14
  SE
       Sódio > 160
          ENTÃO
          Diagnóstico = Hipernatremia Muito Grave

REGRA 15
  SE
       Potássio < 2.5
          ENTÃO
          Diagnóstico = Hipopotassemia Grave

REGRA 16
  SE
       Potássio >= 2,5     < 3
          ENTÃO
          Diagnóstico = Hipopotassemia Importante

REGRA 17
  SE
       Potássio >= 3     < 3,5
          ENTÃO
          Diagnóstico = Hipopotassemia Moderada

REGRA 18
  SE
       Potássio >= 3.5     <= 4.5
          ENTÃO
          Diagnóstico = Potássio Normal

REGRA 19
  SE
       Potássio > 4.5     <= 6
          ENTÃO
          Diagnóstico = Hiperpotassemia Discreta

REGRA 20
  SE
       Potássio > 6     <= 7
          ENTÃO
          Diagnóstico = Hiperpotassemia Grave

REGRA 21
  SE
       Potássio > 7
          ENTÃO
          Diagnóstico = Hiperpotassemia Muito Grave
```

Estas regras serão construídas seguindo o modelo (incremento do número da regra a partir deste valor):

```
  Regra 89
   SE Interface = Diagnóstico Laboratorial
   E  Dosagem de Potássio < 2.5
   ENTÃO Resultado = Hipopotassemia Grave
```

Finalmente todas as variáveis do sistema com suas propriedades:

VARIÁVEIS

```
 Acidose
   Valores:
     Respiratória
```

Metabólica
Tipo:
multivalorada
Alteração Hormonal e/ou metabólica
Valores:
Hipoproteinemia
Doença de Addison
Uso de Insulina + Glicose
Cetoacidose diabética, Mucovicidose, Insuficiência Supra-Renal
Hiperaldosteronismo Primário
Distúrbios do Hormônio Anti-Diurético (Diabetes incididos)
Tipo:
multivalorada
Anorexia
Tipo:
univalorada
Administração Excessiva de Sais
Valores:
Cloro
Potássio
Sódio
Tipo:
multivalorada
Administração de Líquidos sem sais
Valores:
Excessiva
Deficiente (Preso, naufrago. coma, disfagia total (Megaesôfago, Câncer de esôfago))
Tipo:
univalorada
Administração de sais
Valores:
Deficiente
Excessiva
Tipo:
univalorada
Agitação
Tipo:
univalorada
Alcalose
Valores:
Respiratória
Metabólica
Tipo:
multivalorada
Alteração do Trato Gastro-Intestinal
Valores:
Anastomose Uretero Intestinal bilateral.
Lavagem intestinal, diarréia e/ou Íleo
Fístulas - Gástrica, Pancreática, Biliar, Intestinal.
Perdas Gástricas (Vômito - Aspiração)
Tipo:
multivalorada
Ansiedade
Tipo:
univalorada
Antecedentes
Valores:
Alterações do Metabolismo Ácido-Básico
Insuficiência Cardíaca
Perdas através da Pele e dos Pulmões
Alterações do trato Gastro-Intestinal
Administração de líquidos sem sais
Nenhum dos Anteriores
Hemorragias
Insuficiência Renal
Doenças Hormonais e/ou Metabólicas
Perdas Urinárias (Poliúria, Uso de Diuréticos)
Administração de sais
Tipo:
multivalorada
Apatia
Tipo:

univalorada
Arritmia
Valores:
Extrassistoles
Ritmo de Galope
Parada cardíaca em diástole
Fibrilação Ventricular
Parada Cardíaca em Sístole
Sem arritmia
Tipo:
univalorada
Aspecto do Paciente
Valores:
Normal
Edemaciado
Tipo:
univalorada
Ausculta Pulmonar
Valores:
Normal
Dispnéia
Estertores
Tipo:
univalorada
Cefaléia
Tipo:
univalorada
Coma
Valores:
Sem Agitação
Não está em coma
Com agitação
Tipo:
univalorada
Confusão Mental e Torpor
Tipo:
univalorada
Constipação, distensão abdominal, íleo
Tipo:
univalorada
Deficiência de sal
Valores:
Cloro
Potássio
Sódio
Tipo:
univalorada
Delírio
Tipo:
univalorada
Derrames cavitários (Ascite, Derrame Pleural)
Tipo:
univalorada
Desorientação
Tipo:
univalorada
Diurese
Valores:
Normal
Aumentada
Diminuída
Tipo:
univalorada
Dosagem de Cloro
Tipo:
numérica
Dosagem de Potássio
Tipo:
numérica
Dosagem de Sódio
Tipo:
numérica

ECG
  Valores:
    Normal
    Complexo QRS alargado
    Onda U elevada
    RR alongado
    Onda T apiculada e simétrica
    Depressão ST
    QRS alto
    Onda T reduzida, aspecto difásico ou invertida
  Tipo:
    univalorada
Estado metabólico
  Valores:
    Alcalose
    Acidose
  Tipo:
    univalorada
Estupor
  Valores:
    Não está com estupor
    Sem lentidão de Respostas
    Com Lentidão de Respostas
  Tipo:
    univalorada
Excitação
  Tipo:
    univalorada
Extremidades Frias
  Tipo:
    univalorada
Febre
  Tipo:
    univalorada
Fontanelas deprimidas
  Tipo:
    univalorada
Freqüência cardíaca
  Valores:
    Taquicardia
    Normal
    Bradicardia
  Tipo:
    univalorada
Hipertensão Intracraniana (Cefaléia, Vômitos, Hipertensão Arterial, Sinal de Babinski)
  Tipo:
    univalorada
Interface
  Valores:
    Diagnóstico Provável
    Diagnóstico Laboratorial
  Tipo:
    univalorada
Irritabilidade
  Tipo:
    univalorada
Lassidão
  Tipo:
    univalorada
Mucosas
  Valores:
    Húmidas
    Secas
    Normal
    Cianose
  Tipo:
    univalorada
Músculo
  Valores:
    Fraqueza
    Espasmos musculares amplos
    Normal

Hipotonia
Tipo:
univalorada
Náuseas
Tipo:
univalorada
Olhos encovados
Tipo:
univalorada
Perdas Através da pele e pulmão
Valores:
Taquipnéia / Asma
Queimaduras
Sudorese intensa
Tipo:
multivalorada
Pressão Arterial
Valores:
Hipertensão arterial
Sem variação
Hipotensão arterial
Tipo:
univalorada
Pressão venosa
Valores:
Aumentada (Jugular Túrgida)
Normal
Diminuída (Veias colabadas)
Tipo:
univalorada
Pulso
Valores:
Cheio
Fino
Normal
Tipo:
univalorada
Quadro Clínico
Valores:
Hipopotassemia
Hiperhidratação extracelular
Hiperhidratação celular
Hipocloremia
Hiponatremia
Desidratação
Hiperpotassemia
Hipercloremia
Hipernatremia
Tipo:
multivalorada
Resultado
Valores:
Hiperhidratação extracelular
Hipopotassemia
Hiperhidratação celular
Hipocloremia
Hiponatremia
Desidratação
Hiperpotassemia
Hipocloremia Moderada
Hipercloremia
Hipocloremia Média
Hipernatremia
Hipercloremia Moderada
Hipercloremia Grave
Hiponatremia Discreta
Hiponatremia Grave
Hipernatremia Considerável
Potássio Normal
Hipopotassemia Importante
Hiperpotassemia Discreta
Hiperpotassemia muito Grave

Cloro Normal
Hipocloremia Grave
Hipercloremia Acentuada
Sódio Normal
Hiponatremia Importante
Hipernatremia Moderada
Hipernatremia muito Grave
Hipopotassemia Moderada
Normal
Hipopotassemia Grave
Hiperpotassemia Grave
Tipo:
multivalorada
Sede
Tipo:
univalorada
Sensibilidade
Valores:
Anestesia
paralisia
Normal
parestesia
Tipo:
univalorada
Tonturas
Tipo:
univalorada
Torpor
Tipo:
univalorada
Turgor e elasticidade da pele
Tipo:
univalorada
Variação de Peso
Valores:
Aumento
Sem alteração
Diminuição
Tipo:
univalorada
Vômitos
Tipo:
Univalorada

PERGUNTAS

Variável:Acidose
Pergunta:"Qual o tipo de acidose?"
Variável:Alteração Hormonal e/ou metabólica
Pergunta:"Qual a alteração hormonal e/ou metabólica apresentada pelo paciente?"
Variável:Anorexia
Pergunta:"O Paciente apresenta anorexia?"
Variável:Administração Excessiva de Sais
Pergunta:"Qual sal foi administrado iatrogenicamente ao paciente?"
Variável:Administração de Líquidos sem sais
Pergunta:"Foi administrado líquidos sem sais(Ex. Soro glicosado) de forma:"
Motivo:" A administração deficiente de líquidos sem sais ocasiona desidratação e diminuição de
"eletrólitos no extracelular (Sódio, Cloro, Potássio) por diminuição do conteúdo.
" A administração excessiva de líquidos sem sais, principalmente em paciente anúricos(Stress,
"pós-operatório, traumas, etc.) provoca diluição do espaço extracelular ocasionado diluição de
"eletrólitos com conseqüente aparecimento de: hiponatemias, hipocloremia e hipopotassemia
"(A diminuição dos eletrólitos portanto ocorre por aumento do continente). O espaço extracelular
"tendo
"sua osmolaridade diminuída propicia entrada de água na célula com conseqüente aparecimento
"de hiperhidratação intracelular.
Variável:Administração de sais
Pergunta:"De que forma esta sendo feita a administração de sais?"
Motivo:" A administração excessiva de sais (Abuso na dieta, reposição excessiva principalmente em alimentação parenteral prolongada, etc.) leva ao aumento da concentração plasmática dos eletrólitos
"que constituem este sal. Exemplo: a administração excessiva de sal de cozinha (Cloreto de sódio) poderá desencadear aumento da concentração extracelular de sódio e de cloro seus constituintes.
" A administração deficiente de sais (Reposição insuficiente; erro na reposição não se levando em considerações perdas por vômitos, diarréias, fistulas,dentro outros; restrição excessiva na

"administração de sais, como por exemplo paciente cardíaco que faz restrição excessiva do consumo de sal de cozinha) poderá diminuir a concentração deste sais no plasmas levando
"conseqüentemente a situações de déficits eletrolíticos.
Variável:Agitação
Pergunta:"O Paciente está agitado?"
Variável:Alcalose
Pergunta:"Qual o tipo de alcalose?"
Variável:Alteração do Trato Gastro-Intestinal
Pergunta:"Qual(is) a(s) Alteração(es) do Trato gastrintestinal apresentada(s) pelo Paciente?"
Variável:Ansiedade
Pergunta:""
Variável:Antecedentes
Pergunta:"Quais os Antecedentes do Paciente?"
Variável:Apatia
Pergunta:"O paciente esta apático?"
Variável:Arritmia
Pergunta:"O Paciente apresenta arritmia? Qual o tipo?"
Variável:Aspecto do Paciente
Pergunta:"Qual o aspecto do paciente?"
Variável:Ausculta Pulmonar
Pergunta:"Como esta à ausculta do paciente?"
Variável:Cefaléia
Pergunta:"O Paciente tem cefaléia?"
Variável:Coma
Pergunta:"O Paciente está em coma?"
Variável:Confusão Mental e Torpor
Pergunta:"O Paciente apresenta confusão mental acompanhada de torpor?"
Variável:Constipação, distensão abdominal, íleo
Pergunta:"O Paciente apresenta Constipação, distensão abdominal e/ou íleo?"
Variável:Deficiência de sal
Pergunta:"Deficiência na administração de:"
Variável:Delírio
Pergunta:"O paciente esta delirando?"
Variável:Derrames cavitários (Ascite, Derrame Pleural)
Pergunta:"O paciente apresenta derrames cavitários (Ascite, Derrame Pleural)?"
Variável:Desorientação
Pergunta:"O Paciente está desorientado?"
Variável:Diurese
Pergunta:"Como esta à diurese?"
Variável:Dosagem de Cloro
Pergunta:"Qual o Valor do Cloro Plasmático?"
Variável:ECG
Pergunta:"Como esta o Eletrocardiograma?"
Variável:Estado metabólico
Pergunta:"Qual a alteração do equilíbrio ácido-básico apresentado pelo paciente?"
Variável:Estupor
Pergunta:"O Paciente esta com estupor?"
Variável:Excitação
Pergunta:"O paciente está excitado?"
Variável:Extremidades Frias
Pergunta:"As extremidades do paciente estão frias?"
Variável:Febre
Pergunta:"O Paciente tem febre?"
Variável:Fontanelas deprimidas
Pergunta:"As fontanelas estão deprimidas? (Válido somente para crianças que as tenham afastadas)"
Variável:Freqüência cardíaca
Pergunta:"Qual o comportamento da freqüência cardíaca?"
Variável:Hipertensão Intracraniana (Cefaléia, Vômitos, Hipertensão Arterial, Sinal de Babinski)
Pergunta:"O paciente apresenta evidências de Hipertensão Intracraniana (Cefaléia, Vômitos, Hipertensão Arterial, Sinal de Babinski)?"
Variável:Interface
Pergunta:"Qual o Procedimento que você deseja realizar?"
Variável:Irritabilidade
Pergunta:"O paciente apresenta irritabilidade?"
Variável:Lassidão
Pergunta:"O Paciente apresenta lassidão?"
Variável:Mucosas
Pergunta:"Como estão as mucosas?"
Variável:Músculo
Pergunta:"Como esta o paciente?"
Variável:Náuseas
Pergunta:"O paciente tem náuseas?"

```
Variável:Olhos encovados
  Pergunta:"Os olhos estão encovados (Olhos fundos)"
Variável:Perdas Através da pele e pulmão
  Pergunta:"Que Tipo de perda pela pele e/ou pulmão o paciente apresenta?"
Variável:Pressão Arterial
  Pergunta:"Qual o comportamento da pressão arterial?"
Variável:Pressão venosa
  Pergunta:"Como está a pressão venosa?"
Variável:Pulso
  Pergunta:"Como está o pulso?"
Variável:Sede
  Pergunta:"O paciente apresenta sede?"
Variável:Sensibilidade
  Pergunta:"Comportamento da sensibilidade da pele:"
Variável:Tonturas
  Pergunta:"O Paciente tem tonturas?"
Variável:Torpor
  Pergunta:"O paciente esta torporoso?"
Variável:Turgor e elasticidade da pele
  Pergunta:"Turgor e elasticidade da pele diminuída?"
Variável:Variação de Peso
  Pergunta:"Variação de peso?"
Variável:Vômitos
  Pergunta:"O Paciente apresenta vômitos?"
```

5.2.2.3 – Montagem do sistema especialista

A montagem do sistema especialista que permite o diagnóstico de distúrbios hidroeletrolíticos foi feita no 'Expert Sinta' de maneira semelhante a já abordada para 'Escolha da melhor dieta para um paciente cirúrgico' vista neste capítulo.

5.3.3 – Arquivo de Ajuda

O Expert Sinta aceita que, a partir do resultado da consulta, quanto o estado meta é atingido, seja estabelecido um link com um arquivo de ajuda do "Windows" (*.hlp) que, contendo informações pertinentes ao assunto representado no banco de conhecimento, permite ao aluno confirmar se aquele resultado é consistente.

Com o arquivo de ajuda, podem-se montar verdadeiros tutoriais que sedimentarão, por parte do aluno, o entendimento do tema que serviu de modelo para confecção do sistema especialista.

O arquivo de ajuda para diagnóstico de distúrbios hidroeletrolíticos foi confeccionado com o software: Help Magician Pro (O leitor poderá interagir com o sistema especialista e acessar o arquivo de ajuda a partir do aplicativo contido no CD anexo).

Capítulo 6
Conclusões e Perspectivas

6.1 Conclusões

A Inteligência Artificial é fracamente representada nas graduações das Universidades brasileiras em computação, sendo no máximo, uma disciplina obrigatória, depois do sexto período com ementa restrita e na maioria das vezes desatualizada. Economicamente ainda incipiente, talvez por falta de demanda ou de profissionais bem formados.

No Brasil existe uma visão "destorcida e incompleta" do que é IA, diferentemente do exterior, onde, ao contrário, instituições como MIT, Stanford, Carnegie Mellon, Berkeley, Imperial College, Cambridge desenvolvem importantes pesquisas na área.

No final dos anos 50 e inicio dos anos 60, os cientistas Newell, Simon, e J. C. Shaw introduziram o processamento simbólico. Surgiram, desta forma, programas de propósitos específicos, que utilizavam preceitos baseados em um domínio restrito do conhecimento, denominados sistemas especialistas.

Apesar do apelo da IA e dos sistemas especialistas, as aplicações desse segmento da informática já passaram por diversos ciclos com maior ou menor aceitação. No final da década de 80, o ciclo entrou em franca fase, evidentemente, de baixa popularidade. Por isso, nesta época, as aplicações começaram a não ser dirigida somente para o conhecimento de um especialista, mas para uma base de conhecimento. Como conseqüência, foi introduzida a figura do engenheiro do conhecimento, que atua como profissional especializado em transpor uma base de informação para um sistema. Por esse motivo, surgiu um movimento na direção de chamar essa nova geração de sistemas especialistas de 'sistemas com base no conhecimento', já que, os mesmos não se limitavam mais ao conhecimento ou raciocínio simbólico de um único especialista. Esses sistemas podem armazenar, recuperar e analisar vastos repertórios de conhecimento e de dados.

A Pontifícia Universidade Católica do Rio de Janeiro desenvolveu, na década de 1980, importantes trabalhos com sistemas especialistas [Ribeiro 1983].

O Laboratório de Inteligência Artificial da Universidade Federal do Ceará desenvolveu um "shell" de sistema especialista – o Expert Sinta –no final da década de 90, cons-

truído a partir de uma linguagem orientada a objeto – o "object pascal" do Delphi – sendo usado praticamente em todas as grandes Universidades brasileiras no ensino e desenvolvimento de sistemas Inteligentes.

Os novos sistemas inteligentes combinam os conceitos atuais de sistemas especialistas com outras tecnologias, tais como: redes neurais, algoritmos genéticos, realidade virtual e multimídia.

Os sistemas especialistas, por outro lado, ainda continuam apresentando problemas e limitações que restringem seu uso notadamente no que se refere:

- À avaliação de desempenho difícil.
- Ao fato de ser difícil extrair conhecimento especialista.
- Só trabalham muito bem em domínios estreitos.
- Engenheiros do conhecimento são raros e caros.
- Transferência de conhecimento está sujeita a um grande número de preconceitos.

Sistemas especialistas, porém, apresentam vantagens inquestionáveis principalmente no que se refere a:

- O conhecimento do sistema especialista durará indefinidamente.
- Sistemas especialistas valem-se do conhecimento de múltiplos peritos estando disponível para trabalhar simultaneamente e continuamente qualquer hora do dia ou noite na resolução de um problema.
- O sistema especialista aumenta a confiabilidade de que uma boa decisão foi tomada.

A construção de um sistema especialista envolve a extração do conhecimento pertinente a um ou mais peritos humanos. Um engenheiro do conhecimento tem o trabalho de extrair este conhecimento e construir a base de conhecimento do sistema especialista [Amaral at al 1993].

Tendo decidido que o problema é adequado, precisa-se extrair o conhecimento do perito e representá-lo usando um "shell" de sistema especialista. O engenheiro do conhecimento atua como um intermediário entre o perito e o sistema especialista. O trabalho do engenheiro de conhecimento envolve a colaboração do(s) "expert"(s) e do(s) usuário(s) final (is).

O engenheiro do conhecimento, dentre os personagens que lidam com sistemas especialistas, talvez seja a figura mais difícil de ser encontrada em nosso meio (acadêmico, industral, comercial, etc).

Peritos, nos mais variados domínios do conhecimento, são encontrados com muito mais facilidade, e a universidade é a maior prova dessa afirmação. Estes peritos transmitem seu conhecimento através de aulas, palestras, livros, jornais e outras formas de divulgação. A extração do conhecimento para montar um sistema especialista, porém, nem sempre é uma empreitada fácil.

Um engenheiro do conhecimento pode construir sistemas especialistas sem a presença obrigatória de um perito, desde que se disponha a pesquisar e adquirir os ensinamentos documentados nas variadas fontes do conhecimento. Dificilmente o perito pode, por si só, construir bancos de conhecimentos, se não tiver noção de inteligência artificial e elaboração de regras de produção ou outra forma de representação (“frames”, “scripts”, etc).

Alguns formatos que procuram traduzir o conhecimento têm a propriedade de isoladamente permitir que um tema seja resumido de maneira adequada e o seu entendimento, por si só, possibilite a compreensão do todo. Fluxogramas de decisão, quanto utilizados na área médica, são exemplos que ilustram bem o que está sendo afirmado. A representação do conhecimento médico sob o formato de fluxogramas permite que uma sucessão de deliberações seja estruturada no formato de um grafo valorado do qual o elemento mais representativo é a árvore de decisão.

A construção de um programa que, a partir do desenho destas estruturas de representação (fluxograma de decisão), permitisse geração de fatos e heurísticas para construção de sistemas especialistas tornam essa empreitada facilitada já que:

- Fluxogramas de decisão são formas de representação do saber abundantemente encontrados em livros, revistas e outras fontes de conhecimento médico. As fontes de conhecimento em cirurgia não fogem a esta regra.
- Baseados nestas estruturas de representação, um usuário, com noção de uso de computadores e interfaces gráficas, pode obter a partir da fonte bruta as variáveis e as regras para construção de um sistema especialista sem a necessidade da presença de peritos e engenheiros do conhecimento.
- Uma ferramenta deste tipo é um auxiliar importante para que o engenheiro do conhecimento possa gerar regras de produção.

Com este propósito, elaborou-se um programa: O 'Sistema Gerador de Regras' que preenche estes pré-requisitos, pois, é possível, a partir do desenho de um grafo valorado, tradução de um fluxograma de decisão, a obtenção de um relatório impresso (contendo todas as variáveis e regras) que permite a construção de sistemas especialista em qualquer "shell" baseado em regras de produção.

Como fluxogramas de decisão não são as únicas formas de representação do conhecimento médico, mostrou-se também, nesta dissertação, como é o trabalho de elaboração de sistemas especialistas a partir de raciocínios heurísticos, elaborou-se por este motivo, um outro sistema especialista que possibilita o 'Diagnóstico de distúrbios hidroeletrolíticos' (tópico abordado na disciplina: Base da Técnica Cirúrgica – Veja o capitulo V).

A estrutura do sistema especialista pode, portanto, perfeitamente ser adaptada para a construção de sistemas tutoriais, proporcionando um grande potencial para a criação de novos ambientes educacionais. Um sistema tutorial não necessita somente do conhecimento do seu domínio, mas também da perspectiva sobre este conhecimento que permita transmiti-la ao estudante adequadamente.

O uso de sistemas especialistas no ensino médico é uma opção a mais para que o aluno possa sedimentar o conhecimento e incrementar o aprendizado, permitindo interação com o computador (mimetizando a relação médico-paciente) e simulando situações as mais diversas sem que nenhum paciente seja colocado em risco.

Ferramentas criadas para extração do conhecimento, dispensando a necessidade de engenheiros do conhecimento, permitem que o próprio aluno possa implementar sistemas especialistas os mais variados, tornando o estudo tarefa mais agradável, mais divertido, quase um jogo de computador.

6.2 Perspectivas

Para que o SRG seja ferramenta de uso genérico, algumas sugestões serão objetos de trabalhos futuros:

- Integração com o "shell" Expert Sinta: para permitir que as variáveis e o conjunto das regras, que atualmente são geradas em um formulário impresso e introduzidas manualmente, sejam automaticamente validadas no próprio "shell" a partir do SGR. O usuário desenharia o fluxograma, sobre o formato de um grafo valorado, e após validação poderia interagir com o sistema no modo de consulta.

O Expert Sinta tem código fonte aberto significando que melhorias poderão ser introduzidas com facilidade, quais sejam:

- Atualização dos componentes que compõem o 'Expert Sinta' para permitir que os mesmos sejam compilados nas versões 4.0 e superiores do 'Delphi' com compatibilidade para arquivos gerados em versões anteriores. Esta atualização permitiria melhorias nos recursos de criação do "shell".
- Modificação do "prompt" de formulação de perguntas para que o mesmo possa permitir, além do formato texto, voz e mesmo imagem, dando ao sistema especialistas características de multimídia.
- Adaptação para que o mesmo suporte consulta via WEB (Internet) permitindo, com isto, interação à distância.

Bibliografia

ABBOD, M. F.; KEYSERLINGK, D. G. VON; LINKENS, D. A.; MAHFOUF, M. : Survey of utilisation of fuzzy technology in Medicine and Healthcare. Fuzzy Sets and Systems 120 (2001) 331–349

ADAMS, M. B.; CONDON, R. E.: Terapia Hidroeletrolítica in: CONDON, R. E. NYHUS, L. M. Manual de terapêutica Cirúrgica, 5ª edição, Rio de Janeiro, Medsi, 1982, p. 187 - 221.

ALBUQUERQUE, G.de C.; BARROS, E. Distúrbios do equilíbrio Hidroeletrolítico, in: BARRETO, S. S. M. Rotinas em Terapia Intensiva, 2 ª edição, Artes Médicas, 1993

AMÂNCIO, A. Controle Hidroeletrolítico em Cirurgia in: BARBOSA, H.; AMÂNCIO, A. Controle Clínico do Paciente Cirúrgico, 4ª Edição, São Paulo, Livraria Atheneu, p. 87 - 137, 1976.

AMARAL,M. B. ; SATOMURA, Y.; HONDA, M.; SATO, T. A design for decision Making: construction and connection of knowledge bases for a diagnostic System in Medicine, Med. Inform. 18(4): 307 - 320, 1993.

AMARAL,M. B. ; SATOMURA, Y.; HONDA, M.; SATO, T. A Phichiatric Diagnostic System Integrating Propalistic and Categorical Reasoning, Meth. Infom Med, 34; 232 - 243, 1995.

AMARAL,M. B. ; SATOMURA, Y.; HONDA, M.; SATO, T. A Phichiatric Diagnostic System Integrating Propalistic and Categorical Reasoning, Yearbook of Medical Informatics, p. 415 - 426, 1996.

ARARIBÓIA, G.: Inteligência Artificial – Um Curso Prático. . Livros Técnicos e Científicos Editora Ltda, 1988.

BARUZZI, A. C. do A.; HIDAL, J. T.; PEREIRA, M. de B.; KASISSKI, N. Distúrbios do Metabolismo do Magnésio e Cálcio. In: KNOBEL, E. Condutas no Paciente Grave, São Paulo, Livraria Atheneu, 1994.

BECK, L. H. Clínicas Médicas da América do Norte - Distúrbios Hidroeletrolítiticos, Rio de Janeiro, Interamericana, Março de 1981.

BELLMAN, R.: An Introduction to Artificial Intelligence - Can Computers Think? San Francisco - CA. Boyd & Fraser - 1978.

BINDA, R. E. Insuficiência Renal e Distúrbios Hidroeletrolíticos. In: SCHARTZ, G. R. ; SAFAR, P.; STONE, J. H.; STOREY, P. B.; WAGNER, D. K. Emegências Médicas, 1ª Edição, Rio de Janeiro, Interamericana. p. 139 - 148, 1982.

BÖHN, G. M.: A Questão da Educação em Informática Médica. Revista Informédica. 2(10): 19 - 20, 1994.

BRASIL, L. M. ; AZEVEDO, F. M. ;BARRETO, J. M. : A hybrid expert system for the diagnosis of epileptic crisis. Artificial Intelligence in Medicine 21 (2001) 227±233.

BRENT, E. Expert System http://www.missouri.edu/~socbrent/estalk.txt, Universidade de Missouri, 1996.

BROBECK, J. R. As Bases Fisiológicas da Prática Médica, 9ª edição, Rio de janeiro, Guanabara Koogan, 1976.

BUCHANAN, B. G, FEIGENBAUM, E. A.: Dendral and Meta-Dendral: Their aplications Dimension. Artificial Intelligence, 11,(1), p. 5-24, 1978.

CAMPOS, E. R. Um Sistema Interativo de Cálculos para Terapia Intensiva. Revista Informédica. 1(1): 5-11,1993.

CARVALHO, A. F. V. de C.: A Mineração de Dados no Marketing, Medicina, Economia, Engenharia e Administração, Ed Érica. 2001

CARVALHO, J. N. de: S3O: um método de busca de similaridades em objetos estruturados. Tese de Mestrado UFPb, 1999.

CARVALHO, T. F. R. de; SABBATINI, R. M. E.; SOUGEY, E. B.; CAETANO, D. Um Sistema Computadorizado de Apoio ao Diagnóstico de Esquizofrenia. Revista Informédica, 3(16): 15 - 18, 1995.

CASTRO, J.L.; CASTRO-SCHEZ, J. J.; ZURITA, J. M.: Learning maximal struture for knowlwdge acquisition in expert systems. Fuzzy Sets and systems 101(1999) 331 342.

CAUSEY, ALISON: Databases and Artificial Intelligence 3 Artificial Intelligence Segment. 1994. http://www.cee.hw.ac.uk/ãlison/ai3notes/ [2002]

CHAIBEN, H. Inteligência Artificial na Educação, Universidade Federal do Paraná, Centro de Computação Eletrônica, http://www.cce.ufpr.br/~hamilton/iaed/iaed.htm

CHARNIAK, E., McDERMOTT, D.:"Introduction to Artificial Intelligence", Addison-Wesley Publishers, Reading, Massachussets, 1985.

CHUNG,S. S.; AGUIAR,A. C. F. Desidratação na Infância. Residência Médica, 12(1): 11 - 21, 1983.

CIMINO, J. J.; CLAYTON, P. D.; HRIPCSAK, G. Knowledge - based Approaches to the Maintenance of a Large Controlled Medical Terminology, Journal of the Medical Informatics Association, 1(1): 35 - 50, Jan-Fev 1994.

COIERA, E. Designing for Decision Support in a Clinical Monitoring Environment, International Conference on Medical Physics and Biomedical Engineering, University of Cyprus, May 1994 http://wwwuk.hpl.hp.com/people/ewc/Ppes/Cyprus/Cyp_v2.doc.html

COIERA,E. ; BAUD,R. ;CONSOLE, J.; CRUZ, J. DURINCK, J.; FRUTIGER, P.; HUCKLENBROICH, P.; RICKARDS, A.; SPITZER, K. The Role of Knowledge Based System in clinical Practice, in: CHRISTENSEN, B. J. P. Knowledge in Health Telematics -The Next Decade, Amsterdam, p. 199 - 203, 1994.

COIERA, E. Artificial Intelligence in Medicine in: Guide to Medical Informatics, The Internet and Telemedicine, 1997 http://www.coiera.com/aimd.htm.

COIERA, E. W.: Inteligência Artificial na Medicina, Informática Médica, 1(4), 1998.

CRUZ, H. M. M. Desvios do Metabolismo da Água e Eletrólitos in: COSSERMELLI, W.; SALDANHA, R. V.; AZUL,L. G. C de S. Terapêutica Clínica, Rio de Janeiro, Guanabara Koogan, 1979, p. 326 - 333.

DONBAL, F. T. Medical Informatics, Cornwall, Hartnolls, 1996.

DUKE JR.,J. H.; MILLER, T. A. Salt and Eletrolyte Problems in: CONDON, R. E.; DeCOOSE, J. J. Surgical Care: A physiologic Approach to Clinical Management, Philadelphia, Lea & Febiger, p. 349 - 369, 1980.

EBRAHIM, R.: Fuzzy logic programming. Fuzzy Sets and Systems 117 (2001) 215 - 230.

FALSARELLA, O. M.; CHAVES, E. O. C.: Sistemas de Informação e Sistemas de Apoio a Decisão, http://www2.people.com.br/sad.htm.

FAINTUCH, J. Alterações Hidroeletrolíticas no pós-operatório, in: RAIA,A. A.; ZERBINI, E. de J. Clínica Cirúrgica Alípio Corrêa Netto, Volume I, 4ª edição, São Paulo, Sarvier, p. 6 - 23.

FIREBAUGH, M. W.: Artificial Intelligence – A Knowledge-Based Approach. PWS-Kent Publish Company,1988.

FONSECA, D.J.; KNAPP, G. M.: A fuzzy scheme for failure mode screening. Fuzzy Sets and Systems 121 (2001) 453–457

FOSCARINI, L. G.; PEDROSO, E. P. Distúrbios Hidroeletrolíticos e Acido-básicos in: LÓPES, M. Emergências Médicas, 2ª edição, Rio de Janeiro, Guanabara Koogan, 1979, p. 302 - 321.

GARIBALDI, J. M.; WESTGATE, J. A.; IFEACHOR, E. C.: The evaluation of an expert system for the analysis of umbilical cord blood. Artificial Intelligence in Medicine 17 (1999) 109–130.

GIARRATANO,J.; RILEY, G. Expert System principles and programming, 2ª edição, Boston, Pws Publishing Company, 1998.

GENARO, S.: Sistemas Especialistas – O Conhecimento Artificial. Livros Técnicos e Científicos Editora Ltda, 1986.

GOLDEMBERG, E. Alterações do Equilíbrio Hídrico, Eletrolítico e Ácido - Básico, 5ª edição, Rio de Janeiro, Guanabara Koogan, 1978.

GÜVENIN, H. A., EMEKSIV, N.: An expert system for the differential diagnosis of erythemato-squamous diseases. Expert Systems with Applications 18 (2000) 43–49.

HARMON, P.; KING, D.: Expert Systems: Artificial Intelligence in Business. Wiley. New York, 1985.

HARPER, H. A. Terapêutica Hídrica e Eletrolítica in: WILSON, J. L. Manual de Cirurgia, 5ª edição, Rio de Janeiro, Guanabara Koogan, 1976, p. 166 - 193.

HAUGELAND, J..: Mind Design: Philosophy, Psychology in Artificial Intelligence. Mit Press, Cambridge, Massachussets, 1985

HAYES-ROTH, F.: Rule-based Systems, communications of the ACM, 28(9), p. 921-932, 1985.

HEREDIA, L. R.: Mapa Rodoviário for Windows 95. Trabalho Final de Algoritmos e Estruturas de Dados II. Pontifícia Universidade Católica do Rio Grande do Sul. 1996

HIDAL, J. T.; LOFTEMBERG, S. A.; SANTOS, B. F. C dos; LAGUDIS, S. Estados Hiperosmolares, in: KNOBEL, E. Condutas no Paciente Grave, São Paulo, Livraria Atheneu, 1994.

HIRANO, S.; HATA, Y.: Fuzzy expert system for foot CT image egmentation. Image and Vision Computing 19 (2001) 207–216

ILHA, J. O. O Registro Clínico Computadorizado no Hospital. Revista Informédica. 1(3): 5 - 8, 1993.

INNOCENT, P. R.; JOHN, R. I.; GARIBADI, J. M.: The fuzzy medical group in the centre for computational Intelligence, Artificial Intelligence in Medicine 21 (2001) 163±170

IVANDIC, M.; HOFMANN, W.; GUDER, W. G.: The use of knowledge-based systems to improve medical knowledge about urine analysis. Clinica Chimica Acta 297 (2000) 251–260

JACKSON, P.: Expert Sistems. Addison-Wesley Publishing Company. 2 ª ed, 1990

JIMÉNEZ, JOSÉ A.: Lógica computacional, Dpto. de Ciências de la Computación e Inteligencia Artificial – Universidad de Sevilla.

JUNIOR, C. J. F.; SANTOS, B. F. C. dos; AKAMINE, N.; KNOBEL, E. Reposição Volêmica. In: KNOBEL, E. Condutas no Paciente Grave, São Paulo, Livraria Atheneu, 1994.

KOWALSKI, Z.; ARENDT, R.;MELER-KAPCIA, M.; ZIELNSKI, S.: An expert system for aided design of ship systems automation. Expert Systems with Applications 20 (2001) 261±266

KEMBER, N. F. Aplicações do Computador na Medicina, 2ª edição, Editora Campus, 1986.

KIESEL, R. Desenvolvendo Sistemas de Inteligência Artificial. Universidade Regional de Blumenal - FURB, 1996.

KOJIMA, T.; SEKIGUSHI, H.; KOBAYASHI, H; NAKAHARA,S. ; OHTANI, S. : An expert system of machining operation planning in Internet environment. Journal of Materials Processing Technology 107 (2000) 160±166.

KOHN, D. E. Conduta Hidroeletrolítica in: DUNAGAN, W. C.; RIDNER, M. L. Manual de Terapêutica Clínica, 26ª edição, Rio de Janeiro, Medsi, 1991, p. 64 - 87.

KRUPP, M. A.; SWWET, N. J.; JAWETZ, E.; BIGLIERI, E. G. ROE, R. L. Manual Médico, 18ª edição, Rio de Janeiro, Guanabara Koogan, 1977, p. 480 - 494.

KURTZWEIL, R.: The age of intelligent Machines. The MIT Press - Cambridge, Massachussets, 1990.

LAURIKKALA, J.; JUHOLA, M.; LAMMI, S.; PENTTINEN, J. ; AUKEE, P. : Analysis of the imputed female urinary incontinence data for the evaluation of expert system parameters. Computers in Biology and Medicine 31 (2001) 239–257

LEVINSKI, N. G. Líquidos e Eletrólitos, in: THORN, G. W.; ADAMS, R. D.; BRAUNWALD, E.; ISSELBACHER, K. J.; PETERSDORF, R. G. Harrison Medicina Interna, 8ª edição, volume I, Rio de Janeiro, Guanabara Koogan, 1980, p. 355 - 366.

LLACH, F.; KAUFMAN, C. E.; PEDERSON, J. A.; CZERWINSKI, A. W. Fluidos e Eletrolítos in: PAPPER, S.; WILLIAMS, G. R. Manual de Assistência Clínica ao Paciente Cirúgico, 2ª edição, Rio de Janeiro, Medsi, 1982, p. 49 - 75.

LLATA, J. R.; SARABIA, E. G.; ORIA, J. P.: Fuzzy expert system with double knowledge base for ultrasonic classification. Expert Systems with Applications 20 (2001) 347±355.

LONGO, M. B.; SMITH JR., R.; POLISTCHUCK, D. Delphi 3 Total - Aplicações para Banco de Dados, Rio de Janeiro, Brasport, 1997.

LUGER, G. F. STUBBLEFIELD, W.A.: Artificial Intelligence: Structures and Strategies for Complex Problem Solving. Benjamin Cummings Publishers, Redwood City, CA, 1993.

LYERLY, H. K; GAYNOR, J. W. The Handbook of Surgical Intensive Care, 3ª edition, St. Louis, Mosby Year Book, 1992, p. 402 - 448.

MAXWELL, M. H.; KLEEMAN, C. R. Clínica das Alterações Hidroeletrolíticas, 3ª Edição, Rio de Janeiro, Guanabara Koogan, 1981

McCARTHY, J. Some Expert System need common Sence, Stanford University http://www-formal.stanford.edu/jmc , 1984.

McKENDRICK, I. J.; GETTINBY, G.; GU, Y; REID, S. W. J.; C.W. REVIE, C.W.: Using a Bayesian belief network to aid differential diagnosis of tropical bovine diseases, Preventive Veterinary Medicine 47 (2000) 141±156.

MILLER, T. A.; DUKE JR, J. H. Manuseio Hidroeletrolítico. In: DUDRICK, S. J. ; BAUE, A. E.; EISEMAN, B.; MACLEAN, L. D.; ROWE, M. I.; SHELDON, G. F. Manual de Cuidados Pré e Pós - Operatório, 3ª Edição, Rio de Janeiro, Interamericana, p. 33 - 56, 1984.

MILNE, R; NICOL, C.; TRAVÉ-MASSUYÈS, L.: Tiger with model based diagnosis: Initial deployment. Knowledge-Based Systems, 14(2001) 213-222.

NILSSON, N. J.: Principles of Artificial Intelligence. Morgan Kaufmann Publishers, Inc, 1980.

NOGUEIRA, J. H. M.; SILVA, R. B. de A. e; ALCÂNTARA, J. F. L.; ANDRADE, R. C. de Expert Sinta: Uma Ferramenta Visual geradora de Sistemas Especialistas, Laboratório de Inteligência Artificial - LIA/UFC-UECE.

NUSSBAUM, M.: ROSAS, R.; PEIRANO, I. , CÁRDENAS, F. : Development of intelligent tutoring systems using knowledge structures. Computers & Education 36 (2001) 15±32.

PAL, K.: An approach to legal reasoning based on a hybrid decision-support system. Expert Systems with Applications 17 (1999) 1–12.

PALOMBO, C. R. : SABBATINI, R. M. E. O Ensino de Programação de Bancos de Dados em Medicina. Revista Informédica, 1(1): 5 -11, 1993.

PITREZ, F. A. B.: Fundamentos de Pré e Pós Operatório, São Paulo, Byk, 1987 p. 121 - 136.

PRITCHARD, P.: Decision Suppot for GPs: Towards a more certain future? Journal of Informatics in Primary Care. September, 3-5, 1995.

RABELO JR., A. ; ROCHA, A. R. ; SOUZA, A. de S. ; XIMENES, A. A. ; LOBO, N.; CARVALHO, D. ; FILHO, J. W. C. S. ; OLIVEIRA, K. M. de ; SOUZA, L. A. de;

WERNECK, V. M. Um Sistema Especialista para Diagnóstico de Cardiopatias Isquêmicas. Revista Informédica, 1(10): 5-11, 1993.

RABELO JR., A. ; ROCHA, A. R., OLIVEIRA, K.; SOUZA, A.; XIMENES, A.; ANDRADE, C.; ONNIS, D.; OLIVAES, I.; LOBO, N.;FERREIRA, N. WERNECH, V. An Expert system for diagnosis of acute myocardial infarction wit ECG analysis, Artificial Intelligence in Medicine, 10: 75 - 92, 1997

RAMOS, O. L. Distúrbios Hidroeletrolíticos. Estado de Coma. Choque. Hipertensão Arterial in: MARCONDES, M.; SUSTOVICH, D. R.; RAMOS, O. L. Propedêutica Médica, 2ª edição, Rio de Janeiro, Guanabara Koogan, 1979 , p. 885 - 933.

REGÉCZY, N.; GÖRÖG, G.; PÁLOCZI, K : Developing an expert system for immunophenotypical diagnosis in immunodeficiency. Age-related reference values of peripheral blood lymphocyte subpopulations in Hungary. Immunology Letters 77 (2001) 47–54.

RIBEIRO, H. da C e S.: Introdução aos Sistemas especialistas. Livros Técnicos e Científicos Editora S.A. Rio de Janeiro, 1983

RICH, ELAINE; KNIGHT, K.: Inteligência Artificial, 2ª edição, São Paulo, Makron Books, 1994.

ROTELLAR, E. ABC dos Transtornos Eletrolíticos, 2ª edição, São Paulo, Livraria Atheneu, 1977.

RUSSEL, S. ; NORVIG, P.: Artificial Intelligence – A Modern Approach. Prentice Hall. Upper Saddler River, New Jerset, 1995.

SABBATINI, R. M. E. Um teste computadorizado para Auto - Avaliação da Depressão Clínica. Revista Informédica, 1(1): 5 - 11, 1993.

SABBATINI, R. M. [A] Uso do Computador no Apoio ao Diagnóstico Médico. Revista Informédica. 1(1):5-11, 1993.

SABBATINI, R. M [B] Problemas Éticos no Uso do Software de Apoio à Decisão Médica, Revista Informédica, 1(1): 5 - 11, 1993.

SABBATINI, R. M. [C] Um Programa para Apoio ao Diagnóstico do Infarto do Miocárdio. Revista Infomédica, 1(2): 12 -14, 1993.

SABBATINI, R. M. E. PERSONA: Um programa Para a Auto - Avaliação de Personalidade Tipo A/B. Revista Informédica, 2(7): 15 -17, 1994.

SABBATINI, R. M. O CD - ROM na Medicina. Revista Informédica, 2(11): 5 - 11.1194.

SANTOS, B. F. C. dos; ANDREL, A. M.; LOFTEMBERG, S. A.; JUNIOR, M. R. Distúrbios da Concentração Plasmática do Sódio. In: KNOBEL, E. Condutas no Paciente Grave, São Paulo, Livraria Atheneu, 1994.

SANTOS, B. F. C. dos; AKAMINE, N.; BARUZZI, A. C. do A. ; JUNIOR, A. J. S. Distúrbios do Metabolismo do Potássio. In: KNOBEL, E. Condutas no Paciente Grave, São Paulo, Livraria Atheneu, 1994.

SAKELLAROPOULOS, G. C.; NIKIFORIDES,G. C.: Prognostic performance of two expert systems based on Bayesian belief networks. Decision Support Systems 27 2000 431–442.

SCHALKOFF, R. J.: Artificial Intelligence: An Engineering Approach. McGraw-Hill editors, New York, NY 1990

SHIRES, G. T. ; CANIZARO, P. C. Terapêutica Hidroeletrolítica e Nutricional do Paciente Cirúrgico. In: SCHARTZ, S. I., LILLEHEI, R. C.; SHIRES, G. T.; SPENCER, F. C.; STORER, E, H. Princípios de Cirurgia, 2ª Edição, Rio de Janeiro, Guanabara Koogan, p. 65 - 98, 1976.

SHIRES, G. T. ; CANIZARO, P. C. Terapêutica Hidroeletrolítica e Nutricional do paciente Cirúrgico. In:SABISTON, D.C.Tratado de Cirurgia. 1ª edição, Rio de Janeiro, Interamericana , p. 90 - 122, 1979.

SHORTLIFFE, E. H.; PERREAULT, E. H.; FAGAN, L. M.; WIEDERHOLD, G. Medical Informatics - Computer Applications in Health Care, Addison Wesley Publishing Company, 1990

SILVA, R. ; PARIZE, M. M. G. NIACIN: Um Programa para o desenvolvimento de Sistemas Especialistas. Revista Informédica 2(11): 3-16, 1995.

SILVA, R. ; ALCÂNTARA, J. F.; HOLANDA, S.; ANDRADE, R. Aplicações baseadas no Expert Sinta, uma Ferramenta para criação de Sistemas Especialistas, Departamento de Computação, Universidade Federal do Ceará.

SRINIVAS, Y.; TIMMONS, W. D.; DURKIN, J. : A comparative study of three expert systems for blood pressure control. Expert Systems with Applications 20 (2001) 267±274.

SZOLOVITS, P.: Artificial Intelligence in Medicine. Westview Press, Boulder, Colorado. 1982.

SZOLOVITS, P.; PAUKER, S. G.: Categorical and Probabilistic Reasoning in Medicine Revisted. Artificial Intelligence 59: 167 - 180, 1993

TSUMOTO, S.: Knowledge discovery in clinical databases and evaluation of discovered knowledge in outpatient clinic. Information Sciences 124 (2000) 125±137

TURING, A. M.: Computing Machinery and Intelligence. Mind. 59: 433-460, 1950.

VILLELA Jr, G. F: Usando o Computador no Ensino Médico. Revista Informédica 1(1): 5 - 11, 1993.

VOLPE, R. M.: SABBATINI, R. M. E.: Aplicações da Multimídia no Ensino Médico. revista Informédica, 2(9): 5 - 12, 1994.

WICHRT, A.: A categorical expert system "Jurassic". Expert Systems with Applications 19 (2000) 149–158

.WIDMAN, L. E.: Expert System in Medicine, http/:amplatz.uokhsc.edu/acc95-expert-systems.html.

WIDMAN, L. E. Sistemas Especialista em Medicina, Informática médica, 1(5), 1998.

WISTON, P. H. Inteligência Artificial, Rio de Janeiro, Livros Técnicos e Cientificos Ltda, 1988.

WISTON, P. H.: Artificial Intelligence. Addison-Wesley Publishers, Reading, Massachussets, 1992.

WINTERS, R. W. Distúrbios Hídricos e Eletrolíticos in: SILVER, H. K.; KEMPE, C. H.; BRUYN, H. B. Manual de Pediatria, 10ª edição, Rio de Janeiro,Guanabara Koogan, 1975, p. 54 - 78.

"Sites" na Internet (Janeiro 2002)

http://www.hw.ac.uk/

http://www.cce.ufsc.br/~unaberta/estaco/especial/especial2.html/

http://www.cs.cmu.edu/Groups/AI/html/faqs/ai/expert/part1/fac-doc-6.html/

http://www.jsc.nasa.gov/stb/STB_homepage.html/

http://www.jsc.nasa.gov/~clips/CLIPS.html

http://www.aiai.ed.ac.uk/~jacs/wxclips/wxclips.html/

http://www.ghgcorp.com/clips/ExpertSystems.html/

http://www.ghgcorp.com/clips/WhatsIsClips.html/

http://www.medg.Ics.mit.edu/

http://www.epm.br/

http://penta.ufrgs.br/gr952/trab1.geren2.html

http://www.upf.tche.br/computacao/trabalhos/trab-1/ia/SistEsp.htm

http://arachinid.cs.cf.ac.uk

http://www.pucpr.br/disciplinas/bioquimica/bclinica/eqhidro.html

http://www.lia.ufc.br

http://uk.hlp.hp.com/people/ewc/list.idx.htm

http://net.cs.utexas.edu/users/ml/uncertain.html

www.ppgia.pucpr.br/~paraiso

www.di.ufpe.br/~jr

http://www.inf.unisinos.br/~renata

http://www.cs.man.ac.uk/~franconi/teaching/1999/3411/

http://www.cs.man.ac.uk/~franconi/teaching/2000/CT481/

http://grial.uc3m.es/~docweb/ia/

http://www.inf.unisinos.br/~osorio/sistadap.html

Anexo I

Material utilizado para extração dos fatos e regras que permitiram elaborar o sistema especialista para diagnóstico de distúrbios hidroeletrolíticos

Sódio

Massa: 23

Necessidade: 60 – 100 mEq/dia.

1 – 1,4 mEq/Kg de peso.

Ingestão/dia: 5 – 10 g de NaCl.

Sódio corporal total do adulto: 3500 mEq.

Vias de eliminação: Renal (90%), Suor, Fezes.

Hiponatremia

1. Classificação

Discreta: 130 – 138 mEq/l.

Moderada: 125 – 135 mEq/l.

Grave: < 125 mEq/l.

2. Etiologia

Falta de administração de sal de cozinha (Cardíacos, etc).

Diluição por administração excessiva de líquidos em anúricos.

Diarréias, vômitos, aspiração gastrintestinal.

Fístulas biliares ou gastro-intestinal.

Sudorese intensa, queimaduras.

Polaciúrias, uso de diuréticos (Mercuriais, Tiazídicos).

3. Quadro Clínico

Fraqueza.

Anorexia.

Náuseas.

Vômitos.

Apatia.

Lassidão.

Cefaléia.

Confusão mental e torpor.

Coma.

Desaparece turgor e elasticidade da pele.

Hipotensão ortostática.

Taquicardia.

Hipotensão arterial.

Pulso fino.

Extremidades frias.

Veias periféricas colabadas (no choque).

4. Diagnóstico diferencial da Hiponatremia

Por diluição	Por diminuição do conteúdo
Antecedentes de sobrecarga aquosa	Antecedentes de perdas de secreções ricas em sódio
Aspecto inchado do doente	Aspecto normal do doente
Tendência à hipertensão	Tendência à hipotensão
Sinais de sobrecarga Esq. (estertores)	Normalidade cardiopulmonar
Edemas eventuais	Ausência de edemas

5. Laboratório

Hemoconcentração (Hematócrito elevado).

Volume globular médio elevado (Água→ dentro da célula).

Sódio menor que 138 mEq/l.

Volume urinário: geralmente é pequeno

Sódio urinário pode estar baixo ou normal.

6. Tratamento

Cálculo do déficit de sódio:

Sódio: Água extracelular * Déficit de sódio/litro.

Via oral: Sal comum, bicarbonato de sódio.

Via parenteral: Solução salina hipertônica, Solução salina isotônica, lactato de sódio, bicarbonato de sódio, etc.

Hipernatremia

1. Classificação

Moderada: 145 – 150 mEq/l.

Considerável: 150 – 160 mEq/l.

Grave: > 160 mEq/l

2. Etiologia

Ingestão deficiente de água (Coma, náufrago, preso, etc.).

Perda de água maior que a de sódio - Sudorese abundante, diurese osmótica (Manitol), vômitos, etc.

Administração excessiva de sais (sódio).

Hiperaldosteronismo primário.

Distúrbios do metabolismo do hormônio antidiurético (Diabetes insipidus, diabetes insipidus nefrogênico).

3. Quadro clínico

Sede.

Secura das mucosas.

Febre.

Alterações mentais.

Delírio.

Coma com agitação.

4. Laboratório

Aumento da pressão osmótica do plasma.

Aumento da taxa de sódio plasmática.

5. Tratamento

Suspensão do fornecimento.

Diluir e eliminar sódio: Soluções glicosadas.

Maior rapidez:

Diálise peritonial.

Hemodiálise.

Cloro

Funções:

Papel digestivo – HCl.

Equilíbrio osmótico.

Equilíbrio ácido – básico.

Entrada: Cloreto de sódio.

Excreção: Renal, suor, secreção gástrica.

Cloro total: 1490 mEq.

Valor plasmático: 100 mEq/l (96 – 105 Meq/l).

Hipocloremia

1. Classificação

Moderada: 90 – 95 mEq/l.

Média: 80 – 90 mEq/l.

Grave: < 80 mEq/l.

2. Etiologia

Falta de ingestão (Dieta prolongada sem sal).

Perdas gástricas (Vômitos, aspiração gástrica).

Nefropatias diuréticas.

Sudorese significativa.

Acidose respiratória, hipoventilação.

Diluição por excesso de líquidos em anúricos.

3. Quadro clínico

Íleo (vômitos agravam a hipocloremia).

Hipotonia vascular (hipotensão arterial) podendo chegar a colapso circulatório.

4. Tratamento

Tratar a causa básica.

Déficit total de cloro = Água extracelular * déficit de cloro por litro.

Na hipocloremia por diluição eliminar água (Não se deve administrar cloro).

Administração de cloro:

Via oral: Alimentos com cloro, sal comum, cloreto de potássio, etc.

Via parenteral: Solução salina hipertônica, soro fisiológico, Cloreto de Potássio, etc.

Hipercloremia

1. Classificação

Moderada: 105 –115 mEq/l.

Acentuada: 115 – 125 mEq/l.

Grave: > 125 mEq/l.

2. Etiologia

Sobrecarga na administração de sais de cloro (stress pós-operatório, doentes renais)

.

Anastomoses uretero-intestinal bilateral.

Alcalose respiratória.

Distúrbios da função glomérulo tubular.

3. Quadro Clínico

Sede.

Espasmos musculares amplos.

Tremores.

Confusão.

Estupor com lentidão e pobreza de respostas.

Febre baixa.

Falta de controle da micção.

Coma.

Lassidão.

4. Tratamento

Diluição e eliminação do cloro.

Suspensão do fornecimento de cloro.

Hidratação (Soro glicosado).

Ação mecânica com enemas para diminuição do contato entre a mucosa intestinal e ureteral visando diminuir a absorção de cloro (Anastomose uretero– intestinal bilateral).

Bicarbonato de sódio (Acidose).

Potássio

Valor plasmático: 4,5 mEq/l (3,5 – 5 mEq/l).

Necessidades diárias: 40 – 100 mEq/dia.(0,9-0,9 mEq/Kg).

Funções:

Anabolizante

Síntese de proteínas.

Transmissão dos impulsos nervosos.

Musculatura: paralisia.

Miocárdio: parada cardíaca.

Relação com metabolismo ácido-básico:

- Aumenta na acidose.

- Diminuí na alcalose.

Manutenção do meio interno.

Hipercalemia

1. Classificação

Discreta: 5 a 6 mEq/l.

Grave: 6 a 7 mEq/l.

Muito grave: >7 mEq/l.

2. Etiologia

Iatrogênica.

Passagem do intracelular para o extracelular (Acidose).

Contração do extracelular.

Problema renal que dificulte a excreção.

Hemólise, queimadura, choque, etc.

Doença de Addisson.

3. Quadro clínico

SNC: Excitação, ansiedade, agitação, torpor, anestesia, parestesia, fraqueza.

Paralisia muscular respiratória.

Cardiovascular: Bradicardia, extrassístoles, fibrilação ventricular, parada cardíaca em diástole.

E.C.G

Onda T apiculada, alta e simétrica.

Complexo QRS alargado.

4. Tratamento

Supressão de ingressos.

Soluções polarizantes.

4 a 5 g de glicose/1 U de insulina.

Alcalinização

Bicarbonato de sódio.

Medicação antagônica

Gluconato de cálcio.

Resinas de troca

Kayexalate

Diálise – Hemodiálise.

Hipocalemia

1. Classificação

Moderada: 3 – 3,5 mEq/l.

Importante: 2,5 –3 mEq/l.

Grave: < 2,5 mEq/l.

2. Etiologia

Fornecimento insuficiente.

Diuréticos.

Laxativos, lavagem intestinal, diarréias, fístulas, etc.

Alcaloses.

Uso de Insulina + Glicose

3. Quadro clínico

Musculatura lisa: Constipação, distensão abdominal, íleo.

Musculatura estriada: Fraqueza, hipotonia, paralisia flácida.

Musculatura cardíaca: Hipotensão, extrasistoles, parada cardíaca em sístole.

Sistema Nervoso: Apatia, desorientação, hipo ou arreflexia, coma

ECG:

Depressão ST

Onda U elevada

QRS alto

RR alongado

Onda T: reduzida, assume aspecto difásico ou inversão.

4. Tratamento

Nunca isolado. Sempre com grande quantidade de líquidos.

Só repor quando:

Hidratação satisfatória.

Boa diurese (> 700 ml/24 h.).

Densidade urinária > 1017.

Não ultrapassar 20 mEq/h, 200 mEq/24 h.

Ideal: 0,2 mEq/Kg/h.

Quando necessário infundir em grande quantidade:

Monitorizar o paciente (E.C.G.).

Potássio sérico 3/3 h.

Água

Hiperidratação extra celular (Edema)

1. Etiologia

Cardíaco.

Renal.

Endócrino.

Inflamatório.

Alérgico.

Hipoproteinemia.

2. Quadro clínico

Manifestação da doença causal (Cardíaco, renal, etc).

Rápido aumento de peso.

Hipertensão venosa.

Dispnéia.

Cianose.

Ritmo de galope.

Derrames serosos, edema subcutâneo.

3. Tratamento

Tratamento da doença causal.

Diuréticos.

Redução do sódio: uso de antagonistas da aldosterona (espirinolactona).

Diálise peritonial com soluções hipertônicas.

Hiper-hidratação celular

1. Etiologia

Administração excessiva de água sem sais:

Durante oligúria por doença renal.

Fase de injúria do pós-operatório.

Sudorese profusa.

No tratamento das perdas de água e sais: diarréia, vômitos, fistulas, etc.

Ingestão excessiva de água:

Por sudorese.

Pacientes psicóticos.

Após enemas de água em crianças portadoras de megacolon.

2. Quadro clínico

Síndrome de hipertensão intracraniana:

Bradicardia, bradpnéia, hipertensão arterial.

Babinski, distúrbios psíquicos.

Náuseas, vômitos.

Edema pulmonar e subcutâneo.

Aumento rápido do peso: sinal importante.

3. Tratamento

Suspensão da administração de água sem eletrólitos.

Administração de solutos hipertônicos (diurese osmótico): glicose, manitol.

Solução hipertônica de cloreto de sódio

6 ml/Kg peso corporal/ 24 h.

Diálise peritonial.

Desidratação

1. Classificação

Extracelular.

Intracelular.

Mista.

2. Etiologia

Privação de líquidos (Neuropatia, coma).

Ingestão deficiente de água (disfagia total: câncer, megaesôfago).

Aspiração gástrica ou intestinal.

Queimaduras.

Doenças metabólicas:

Mucovicidose.

Cetoacidose diabética.

Insuficiência supra-renal aguda.

Hemorragias.

Administração excessiva de líquidos hipertônicos (diurese osmótico).

3. Quadro clínico

Perda de peso.

Olhos encovados.

Fontanela deprimida.

Mucosas secas, língua saburrosa.

Pele seca com elasticidade diminuída e turgor pastoso.

Sede. Oligúria.

Irritabilidade aumentada.

Hemoconcentração (Hematócrito e leucócitos aumentados).

Hemodiluição só nas hemorragias (Hematócrito diminuído).

4. Grau da desidratação

Moderada: perda de 5 a 10% do total de água corpóreo.

Média: perda de 10 – 20% do total de água corpóreo.

Grave: perda de mais de 20% do total de água corpóreo.

5. Tipos de desidratação

Osmolaridade normal: 280 – 310 mOs/l.

Osmolaridade ≅ (2 * Sódio plasmático) + 10.

Desidratação isotônica.

Desidratação hipertônica.

Desidratação hipotônica.

6. Tratamento

Acompanhamento do peso.

Requerimento basal + Déficit estático.

Anexo II

Fonte

```
{*************************************************************************
 *                    UNIVERSIDADE FEDERAL DA PARAÍBA                    *
 *                       UNIVERSIDADE TIRADENTES                         *
 *************************************************************************
 *                          Mestrado em Informática                      *
 *               Sistema Especialista como auxiliar para                 *
 *               o Ensino de Bases da Técnica Cirúrgica                  *
 *     "Gerador de Regras de Produção a partir de Grafos Valorados"      *
 *                                                                       *
 *                                                                       *
 *   Orientador: Antonio Ramirez Hidalgo                                 *
 *   Mestrando: João Fernandes Britto Aragão                             *
 *                                                                       *
 *   Data: Janeiro de 2002                                               *
*************************************************************************}

unit fonte1;

interface

uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dia-
logs,
  ExtCtrls, StdCtrls, Buttons, Menus, ComCtrls, Grids, Mask,Fonte2;

const TAMNODO=20; {Define o tamanho do diâmetro da circunferência que é
desenhada no mapa para representar cada nodo, isto é, cada nodo
é uma circunferência  com centro em (xiVar,yiVar) e raio 10}
      SELECIONA=0; {Constante utilizada na função ZonaCritica, se ela for
passada é sinal que o usuário está selecionando um nodo, ver Função Zona-
Critica}

type
  str100=string[100]; {Usadas para possibilitar a passagem desses tipos}
  str2=string[2];   {por parâmetro}
  TipoVariavel = (univalorada,multivalorada,numerica);
  PTRListaAdj=^ListaAdj;{ Ponteiro para Lista Adjacente}
  {Definição da Lista principal do Grafo}
  PTRGrafo=^Grafo; {Ponteiro para a estrutura Grafo}
   Grafo=record
         CodNodo:str2;
         Nome_nodo:str100;
         Opcao_tipo:Tipovariavel;
         Objetivo:boolean;
         Pergunta: string;
         Motivo:string;
         relacionamento1:str2;
         Nome_valor1:str100;
         relacionamento2:str2; {Quando faz parte de um intervalo numérico}
         Nome_valor2:str100;   {Quando faz parte de um intervalo numérico}
         xiNodo,yiNodo,xfNodo,yfNodo:integer; {Coordenadas que posicionam
```

```
o nodo no mapa}
        Prox:PTRGrafo; {Para efetuar o encadeamento}
        PrimAdj:PTRListaAdj; {Aponta para o primeiro elemento da lista de
adjacência}
        UltAdj:PTRListaAdj;  {Aponta para o último elemento da lista de
adjacência}
        end;
  {Definição da estrutura ListaAdj}
  {Essa é a lista de nodos adjacentes de cada nodo da estrutura Grafo}
  ListaAdj=record
        Nodo:PTRGrafo; {Aponta para o elemento do grafo}
        Relacionamento1:str2;
        nome_valor1:str100; {Armazena o valor de um nodo origem que per-
mite o aparecimento do nodo adjacente - Em IA é a estrutura que permitirá
a movimentação em Espaços de Busca junto com relacionamento1}
        Relacionamento2:str2;
        nome_valor2:str100; {Quando faz parte de um intervalo numérico}
        Prox:PTRListaAdj; {Aponta para o próximo da lista}
        Ant:PTRListaAdj;  {Aponta para o anterior da lista (lista dupla-
mente encadeada}
        end;
  PTRListaOrigem=^ListaOrigem;{Ponteiro para Lista Origem}
  {Definição da Lista de Origem - Todos os nós que são nodos fontes}
  ListaOrigem=record
        NodoOrigem:PTRGrafo;
        Prox:PTRListaOrigem;{Aponta para o próximo da lista}
        end;
  PTRListadestino=^ListaDestino;{Ponteiro para Lista Destino}
  {Defenição da Lista de Destino - Todos os nós que são nodos objetivos}
  ListaDestino=record
        NodoDestino:PTRGrafo;
        Prox:PTRListaDestino; {Aponta para o próximo da lista}
        end;

  {Definição da estrutura PilhaRegra}
  {Essa estrutura é utilizada no algoritmo de caminhamento no grafo.É um
caminhamento em profundidade, por isso necessita dessa Pilha Auxiliar. No
final de cada caminhamento o conteúdo dessa pilha é o caminho encontrado
para a formação das regras}
  PTRPilhaRegra=^PilhaRegra;
  PilhaRegra=record
        Nodo:PTRGrafo; {Aponta para um nodo no grafo}
        relacionamento1:str2;{Armazena o sinal de atribuição de valor}
        nome_valor1:str100; {Armazena o valor da variável}
        relacionamento2:str2;{Armazena o sinal de atribuição de valor -
Intervalo}
        nome_valor2:str100; {Armazena o valor da variável -Intervalo}
        Prox:PTRPilhaRegra; {Para poder encadear}
        end;

  {Definição da estrutura Lista Regras}
  {Estrutura que armazena todos os caminhos encontrados num caminhamento
   entre um nó fonte no mapa grafo (Origem) e o que seria o sumidouro
(Destino). Os resultados são armazenados de forma tabular, utilizando
como índice a variável linha.
   Exemplo: Caso os caminhos encontrados entre a variável A e D fossem:
            Se A e B e C então D,Se A e C então D, Se A então D. Esses
estariam armazenados da seguinte forma:
     Linha   Nodo
```

```
      1       A
      1       B
      1       C
      1       D
      2       A
      2       C
      2       D
      3       A
      3       D
      ----------}
  PTRListaRegra=^ListaRegra;
  ListaRegra=record
        Nodo:PTRGrafo; {Aponta para o nodo do grafo}
        Linha:LongInt; {Contém a linha no qual o ponteiro é armazenado}
        Prox:PTRListaRegra; {Para realizar o encadeamento}
        end;

  {Definição do arquivo Tipado Arquivo}
  {Arquivo tipado (binário) que armazena em cada registro uma estrutura
do
   tipo grafo}
  Arquivo=File of Grafo;

  {Definição da classe TForm1}
    TFrmMapa = class(TForm)
    Mapa: TPaintBox;
    MainMenu1: TMainMenu;
    Arquivo1: TMenuItem;
    Novo: TMenuItem;
    Abrir: TMenuItem;
    Salvar1: TMenuItem;
    Imprimir: TMenuItem;
    Imprimir1: TMenuItem;
    Sair1: TMenuItem;
    AbrirDialog: TOpenDialog;
    SalvarDialog: TSaveDialog;
    BtnResultado: TBitBtn;
    ImprimirDialog: TPrintDialog;
    PageMapa: TPageControl;
    Tabnodo: TTabSheet;
    TabValor: TTabSheet;
    CmbGrafo: TComboBox;
    EdCodNodo: TMaskEdit;
    Label3: TLabel;
    Label4: TLabel;
    EdNomeNodo: TEdit;
    BtnInclui: TBitBtn;
    BtnExclui: TBitBtn;
    BtnAltera: TBitBtn;
    BtnLimpa: TBitBtn;
    GroupBox1: TGroupBox;
    CmbOrigem: TComboBox;
    GroupBox2: TGroupBox;
    Cmbdestino: TComboBox;
    GroupBox3: TGroupBox;
    GroupBox4: TGroupBox;
    RadioSim: TRadioButton;
    RadioNao: TRadioButton;
    CmbRelacionamento1: TComboBox;
```

```
    BtnAplDist: TBitBtn;
    BtnExc: TBitBtn;
    RGObjetivo: TRadioGroup;
    RGTipo: TRadioGroup;
    EdPergunta: TEdit;
    EdMotivo: TEdit;
    LabelPergunta: TLabel;
    LabelMotivo: TLabel;
    EdValor1: TEdit;
    BitSaida: TBitBtn;
    RGIntervalo: TRadioGroup;
    EdValor2: TEdit;
    CmbRelacionamento2: TComboBox;
    procedure MapaMouseDown(Sender: TObject; Button: TMouseButton;
      Shift: TShiftState; X, Y: Integer);
    procedure AtualizaCmbGrafo;
    procedure CmbGrafoClick(Sender: TObject);
    procedure ConsultaGrafo(PrimGrafo:PTRGrafo;Chave:str100);
    procedure AplDadosGrafo(var PAux:PTRGrafo;RegGrafo:Grafo);
    procedure IncluiGrafo(var PrimGrafo:PTRGrafo;
              var UltGrafo:PTRGrafo;RegGrafo:Grafo);
    procedure DesNodo(x,y:integer;Codigo:str2);
    procedure DesMapa;
    procedure MapaPaint(Sender: TObject);
    procedure ZonaCritica(novoxi,novoyi,novoxf,novoyf,operacao:integer;
              var RetLogic:integer;var RetPTR:PTRGrafo);
    procedure BtnIncluiClick(Sender: TObject);
    procedure MapaMouseUp(Sender: TObject; Button: TMouseButton;
      Shift: TShiftState; X, Y: Integer);
    procedure LimpaMapa;
    function InGrafo(PrimGrafo:PTRGrafo;CodNodo:str2):boolean;
    procedure AtuJanNodo;
    procedure FormCreate(Sender: TObject);
    procedure BtnAlteraClick(Sender: TObject);
    procedure BtnExcluiClick(Sender: TObject);
    procedure LimpaFormNodo;
    procedure BtnAplDistClick(Sender: TObject);
    function  AtribPtrGrafo(PrimGrafo:PTRGrafo;Chave:str100):PTRGrafo;
    procedure AtribDistGra-
fo(POrigem,PDestino:PTRGrafo;relacionamento1:str2;
              no-
me_valor1:str100;relacionamento2:str2;nome_valor2:str100);
    procedure CmbOrigemChange(Sender: TObject);
    procedure ExibeValor_Nodo(PrimGrafo,POrigem,PDestino:PTRGrafo);
    procedure CmbDestinoChange(Sender: TObject);
    procedure BtnExcClick(Sender: TObject);
    procedure ExcluiCaminho(var PrimAdj:PTRListaAdj;var Ul-
tAdj:PTRListaAdj;
                            var PAuxAdj:PTRListaAdj);
    procedure ExcluiLigacoes(PAtualGrafo:PTRGrafo);
    procedure ConstruirRegras(POrigem,PDestino:PTRGrafo);
    procedure Empilha(Nodo:PTRGrafo);
    procedure Desempilha;
    function Visitada(Nodo:PTRGrafo):Boolean;
    procedure AtuJanRegras;
    procedure IncListaRegra(var PrimListaRegras:PTRListaRegra;
      var UltListaRegra:PTRListaRegra;linha:longint;Nodo:PTRGrafo);
    procedure LimpaListaRegra(var PrimListaRegras:PTRListaRegra;
      var UltListaRegra:PTRListaRegra);
```

```
    procedure Salvar1Click(Sender: TObject);
    procedure Salvar(NomeArq:string);
    procedure AbrirClick(Sender: TObject);
    procedure AbrirArq(NomeArq:string);
    procedure LimpaGrafo(var PrimGrafo:PTRGrafo;var UltGrafo:PTRGrafo);
    procedure NovoClick(Sender: TObject);
    function  AtribPtrGrafoCod(PrimGrafo:PTRGrafo;Chave:str2):PTRGrafo;
    procedure ImprimirClick(Sender: TObject);
    procedure BtnLimpaClick(Sender: TObject);
    procedure Sair1Click(Sender: TObject);
    procedure CmbOrigemCamChange(Sender: TObject);
    procedure CmbDestinoCamChange(Sender: TObject);
    procedure BtnResultadoClick(Sender: TObject);
    procedure Saida;
    procedure FormClose(Sender: TObject; var Action: TCloseAction);
    procedure RGObjetivoClick(Sender: TObject);
    procedure IncListaOrigem(var PrimListaOrigem:PTRListaOrigem;
          var UltListaOrigem:PTRListaOrigem;Nodo:PTRGrafo);
    procedure IncListaDestino(var PrimListaDestino:PTRListaDestino;
          var UltListaDestino:PTRListaDestino;Nodo:PTRGrafo);
    Procedure FormaRegra;
    Procedure FormarListaOrigem;
    Procedure FormarListaDestino;
    procedure VariaveisDoSistema;
    procedure DefinicaoObjetivo;
    Function EstaNosAdjacentes(var Nodo:PTRGrafo):boolean;
    procedure LimpaListaOrigem(var PrimListaOrigem:PTRListaOrigem;
          var UltListaOrigem:PTRListaOrigem);
    procedure LimpaListaDestino(var PrimListaDestino:PTRListaDestino;
          var UltListaDestino:PTRListaDestino);
    procedure DesNodoObjetivo(x,y:integer;Codigo:str2);
    procedure BitSaidaClick(Sender: TObject);
    procedure RGIntervaloClick(Sender: TObject);
    procedure EdValor1KeyPress(Sender: TObject; var Key: Char);
    procedure EdValor2KeyPress(Sender: TObject; var Key: Char);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
{Variáveis globais do programa}
  FrmMapa: TFrmMapa;
  Resultado:string; {Nome da variável objetivo}
  TipoObjetivo:string;{Recebe o Tipo da variável objetivo}
  Movendo:Boolean; {Variável que tem o seu valor verdadeiro se o usuário
estiver movendo um círculo que representa um nó no mapa}
  PrimGrafo,UltGrafo:PTRGrafo; {Ponteiros para o Primeiro e Últimos ele-
mentos do grafo}
  RegGrafo:Grafo; {Variável da estrutura Grafo, utilizada para atribuir
valores ao grafo}
  PAtualGrafo:PTRGrafo; {Ponteiro para o Elemento Atual do grafo.
Elemento selecionado atualmente, é sobre ele que as funções serão execu-
tadas.}
  PrimListaOrigem,UltListaOrigem:PTRListaOrigem; {Ponteiros para o pri-
meiro e ultimo elemento da estrutura ListaOrigem}
  PrimListaDestino,UltListaDestino:PTRListaDestino; {Ponteiros para o
primeiro e ultimo elemento da estrutura ListaDestino}
```

```
  Topo:PTRPilhaRegra; {Ponteiro para o Topo da Estrutura PilhaRegra}
  PrimListaRegra,UltListaRegra:PTRListaRegra; {Ponteiros para o primeiro
e últimos elementos da estrutura ListaRegra}
  NomeArq:string; {Variável que armazena o nome do arquivo aberto}
  nome_valor1:str100;
  Relacionamento1 : str2;
  nome_valor2:str100;
  Relacionamento2 : str2;
  Numero:Integer;  {Número de Ordem da regra}
implementation

{$R *.DFM}

{=================================================================
 =================================================================
                        INICIALIZAÇÕES E FINALIZAÇÕES
 =================================================================
 =================================================================}

{Executado quando o FrmMapa é criado}
procedure TFrmMapa.FormCreate(Sender: TObject);
begin
  {Inicializa todos os ponteiros}
  PAtualGrafo:=nil;
  PrimListaOrigem:=nil;
  UltListaOrigem:=nil;
  PrimListaDestino:=nil;
  UltListaDestino:=nil;
  Topo:=nil;
  PrimListaRegra:=nil;
  UltListaRegra:=nil;
  { Para que intervalo fique invisível}
  RGIntervalo.Visible := false;
  CmbRelacionamento2.Visible :=false;
  EdValor2.Visible :=false;
   {Coloca o sinal de relacionamento para um valor atribuído a uma aresta
do grafo - Default - '=' }
  CmbRelacionamento1.Items.Add('=');
  CmbRelacionamento1.Items.Add('<>');
  CmbRelacionamento1.ItemIndex :=0;
end;

{Limpa o grafo todo, e também todos os objetos. Executado quando um novo
 arquivo é aberto, ou quando o usuário seleciona a opção Novo no menu.
 Executado também quando o programa é encerrado}
procedure TFrmMapa.LimpaGrafo(var PrimGrafo:PTRGrafo;var UltGra-
fo:PTRGrafo);
var PAux:PTRGrafo;
begin
  PAux:=PrimGrafo;
  while PAux<>nil do begin
        PAux:=PrimGrafo;
        PAux^.Prox:=nil;
        Dispose(PAux);
        PAux:=PrimGrafo^.Prox;
        PrimGrafo:=PAux;
        UltGrafo:=PAux;
  end;
  CmbOrigem.Clear;
```

```
  CmbDestino.Clear;
  CmbGrafo.Clear;
  EdCodNodo.Clear;
  EdNomeNodo.Clear;
  EdPergunta.Clear;
  EdMotivo.Clear;

end;

{Executado quando a janela do programa é restaurada}
procedure TFrmMapa.MapaPaint(Sender: TObject);
begin
  DesMapa;
end;

{Executado quando se desejar sair do programa}
procedure TFrmMapa.Saida;
begin
  {Pergunta se não deseja salvar o arquivo}
  if MessageDlg('Deseja Salvar o Mapa em Arqui-
vo',mtConfirmation,[mbYes,mbNo],0)=idYes
     then begin
     SalvarDialog.FileName:=NomeArq;
     if SalvarDialog.Execute then begin
        NomeArq:=SalvarDialog.FileName;
        Salvar(NomeArq);
     end;
     end
  else begin
     LimpaGrafo(PrimGrafo,UltGrafo);
     LimpaListaOrigem(PrimListaOrigem,UltListaOrigem);
     LimpaListaDestino(PrimListaDestino,UltListaDestino);
     LimpaListaRegra(PrimListaRegra,UltListaRegra);
     DesMapa;
  end;
end;

{Executado quando FrmMapa é fechado}
procedure TFrmMapa.FormClose(Sender: TObject; var Action: TCloseAction);
begin
  Saida; {Executa o procedimento de saída}
end;

{=============================================================================
 =============================================================================
                        OPERAÇOES NO FORMULÁRIO
 =============================================================================
 =============================================================================}

 {Limpa o formulário de edição dos Nodos}
procedure TFrmMapa.LimpaFormNodo;
begin
  EdCodNodo.Text:='';
  EdNomeNodo.Text:='';
  EdPergunta.Text := '';
  EdMotivo.Text := '';
  CmbGrafo.Text:='';
  RGTipo.ItemIndex:=0;
  RGObjetivo.ItemIndex:=1;
```

```
  EdCodNodo.SetFocus ;
 end;
{=============================================================================
 =============================================================================
                          DESENHO DO GRAFO
 =============================================================================
 =============================================================================}

{Desenha o nodo no mapa}
procedure TFrmMapa.DesNodo(x,y:integer;Codigo:str2);
begin
  Mapa.Canvas.Pen.Width:=1; {Faz a linha ter espessura 1}
  Mapa.Canvas.Pen.Color:=clBlue; {Cor da linha azul}
  Mapa.Canvas.Ellipse(X,Y,X+TAMNODO,Y+TAMNODO);
                    {Desenha a elipse de diâmetro TAMNODO}
  Mapa.Canvas.TextOut(X+6,Y+3,Codigo);
                    {Escreve o código do nó dentro da elipse}
end;

{Desenha o nodo objetivo no mapa}
procedure TFrmMapa.DesNodoObjetivo(x,y:integer;Codigo:str2);
begin
  Mapa.Canvas.Pen.Width:=1; {Faz a linha ter espessura 1}
  Mapa.Canvas.Pen.Color:=clBlack; {Cor da linha preta}
  Mapa.Canvas.Rectangle(X,Y,X+TAMNODO,Y+TAMNODO);
                    {Desenha o quadrado de diâmetro TAMNODO}
  Mapa.Canvas.TextOut(X+6,Y+3,Codigo);
                    {Escreve o código do nó dentro do quadrado}
end;

{Desenha o mapa completo - Todas os nodos e suas ligações}
procedure TFrmMapa.DesMapa;
var PAuxGrafo:PTRGrafo;
    PAuxAdj:PTRListaAdj;
    xiorigem,yiorigem,xidest,yidest,xidiv,yidiv,k:integer;
begin
  PAuxGrafo:=PrimGrafo;
  LimpaMapa; {Limpa o mapa}
  k:=-3; {Razão de divisão igual a -3, utilizado para dividir o segmento
que liga um nó a outro.Com essa divisão uma parte final do segmento terá
uma cor diferente, identificando que o nodo que receber a linha dessa cor
diferente é o nodo destino}
  {Primeiro Desenha todas as linhas}
  while(PAuxGrafo<>nil) do begin
       PAuxAdj:=PAuxGrafo^.PrimAdj;
       while(PAuxAdj<>nil) do begin
            {Atribui os valores das coordenadas}
            xiorigem:=PAuxGrafo^.xiNodo+10;
            yiorigem:=PAuxGrafo^.yiNodo+10;
            xidest:=PAuxAdj^.Nodo^.xiNodo+10;
            yidest:=PAuxAdj^.Nodo^.yiNodo+10;
            {Divide o segmento numa razão - Para poder identificar quem é
             a origem e quem é o destino}
            xidiv:=round((xiorigem-(k*xidest))/(1-k));
            {Ponto que Divide um segmento numa razão k}
            {** Geometria Analítica!!!}
            yidiv:=round((yiorigem-(k*yidest))/(1-k));
            {Desenha a linha que liga os dois}
            Mapa.Canvas.MoveTo(xiorigem,yiorigem);
```

```
            Mapa.Canvas.LineTo(xidest,yidest);
            {Muda a cor e desenha a parte do segmento que identificará
quem é o destino}
            Mapa.Canvas.Pen.Color:=clRed;
            Mapa.Canvas.Pen.Width:=2;
            Mapa.Canvas.LineTo(xidiv,yidiv);
            Mapa.Canvas.Pen.Color:=clBlue;
            Mapa.Canvas.Pen.Width:=1;
            PAuxAdj:=PAuxAdj^.Prox;
       end;
       PAuxGrafo:=PAuxGrafo^.Prox;
  end;
  {Desenha os ícones dos nodos}
  PAuxGrafo:=PrimGrafo;
  while(PAuxGrafo<>nil) do begin
     if PAuxgrafo^.Objetivo =True then
       DesNodoObjeti-
vo(PAuxGrafo^.xiNodo,PAuxGrafo^.yiNodo,PAuxGrafo^.CodNodo)else
       DesNodo(PAuxGrafo^.xiNodo,PAuxGrafo^.yiNodo,PAuxGrafo^.CodNodo);
       PAuxGrafo:=PAuxGrafo^.Prox;
  end;
end;

  {Procedimento ZonaCritica
      Parâmetros: novoxi,novoyi,novoxf,novoyf - identificam as coordenada
do mapa onde se deseja pesquisar se há ou não um nó nessa região.
      operação - identifica a operação que está sendo realizada
Se a operação for de selecionar um nó (SELECIONA), atribui ao ponteiro
RetPtr o ponteiro para o nó encontrado.
      RetLogic - Recebe valor 0 se não foi encontrado nenhum nó.
      RetPtr - Se a operacao for SELECIONA então esse ponteiro aponta pa-
ra o ponteiro do nó encontrado, se não for encontrado nenhum nodo nessa
região, aponta para NULL}
procedure
TFrmMapa.ZonaCritica(novoxi,novoyi,novoxf,novoyf,operacao:integer;
          var RetLogic:integer;var RetPTR:PTRGrafo);
var PAux:PTRGrafo; {Ponteiro auxiliar para o Grafo}
    PosCritica1,PosCritica2:Boolean;
begin
  PAux:=PrimGrafo; {Para iniciar no começo do grafo}
  PosCritica1:=False;
  PosCritica2:=False;
  while(PAux<>nil) do begin
       PosCritica1:=False;
       PosCritica2:=False;
       {Verifica se as coordenadas recebidas pela função estiverem entre
um dos nodos, Se estiverem então PosCritica1 ou 2 recebe TRUE}
       if ((novoxi>=PAux^.xiNodo) and (novoxi<=PAux^.xfNodo)) or
          ((novoxf>=PAux^.xiNodo) and (novoxf<=PAux^.xfNodo)) then
          {Verifica no eixo x}
           PosCritica1:=True;
       if ((novoyi>=PAux^.yiNodo) and (novoyi<=PAux^.yfNodo)) or
          ((novoyf>=PAux^.yiNodo) and (novoyf<=PAux^.yfNodo)) then
          {Verifica no eixo y}
           PosCritica2:=True;
       if (PosCritica1) and (PosCritica2) then break; {Se estiver no eixo
x e no eixo y então é sinal que já existe um nó nesse local. Interrompe a
pesquisa nesse instante}
       PAux:=PAux^.Prox;
```

```
  end;
if operacao=SELECIONA then begin
   if (PosCritica1) and (PosCritica2) then RetLogic:=1 {Retorna 1 se foi
encontrado algum nó nessa região}
   else RetLogic:=0; {Retorna 0 se não foi encontrado nada}
   RetPTR:=PAux; {RetPTR recebe o PAux atual}
end;
end;

{Procedimento executado quando o botão do mouse é pressionado sobre o
mapa. Se existir algum nodo na posição clicada, então o ponteiro PAtual
aponta para esse nodo. Para pesquisar se há ou não nodo nesse espaço é
utilizado o Procedimento ZonaCritica}
procedure TFrmMapa.MapaMouseDown(Sender: TObject; Button: TMouseButton;
  Shift: TShiftState; X, Y: Integer);
var
  RetLogic:integer; {Valor retornado pela função ZonaCrítica
                     Retorna 0 se não foi encontrado nenhum nodo}
  RetPTR:PTRGrafo; {Ponteiro retornado pela função ZonaCritica Retorna
apontando para o endereço onde esta localizado o nodo selecionado no ma-
pa}
begin
  PageMapa.ActivePage:=TabNodo; {Ativa a página Nodo do objeto PageCon-
trol}
  ZonaCritica(x,y,x+TAMNODO,y+TAMNODO,SELECIONA,Retlogic,RetPTR);
  if RetLogic=0 then exit;
  PAtualGrafo:=RetPtr; {Ponteiro do Atual aponta para o ponteiro retorna-
do pela ZonaCritica}
  AtuJanNodo; {Atualiza a janela de dados sobre o nodo}
  Movendo:=True; {Movendo recebe True}
end;

procedure TFrmMapa.MapaMouseUp(Sender: TObject; Button: TMouseButton;
  Shift: TShiftState; X, Y: Integer);
var
  RetLogic:integer;
  RetPTR:PTRGrafo;
begin
  if not Movendo then exit;
  Movendo:=False;
  if (x<0) or (y<0) or (x>(Mapa.Left+Mapa.Width-20)) or
     (y>(Mapa.Top+Mapa.Height-20)) then begin
     ShowMessage('O Nodo não pode ser incluída nessa posição.');
     exit;
  end;
  ZonaCritica(x,y,x+TAMNODO,y+TAMNODO,SELECIONA,Retlogic,RetPTR);
  if RetLogic=1 then exit;
  PAtualGrafo^.xiNodo:=x;
  PAtualGrafo^.yiNodo:=y;
  PAtualGrafo^.xfNodo:=x+TAMNODO;
  PAtualGrafo^.yfNodo:=y+TAMNODO;
  DesMapa;
end;

{Limpa o mapa}
procedure TFrmMapa.LimpaMapa;
begin
  with Mapa do begin
       Canvas.Pen.Color:=clBlue;
```

```
        Canvas.Brush.Color:=clWhite;
        Canvas.Rectangle(0,0,Width,Height);{Desenha um retângulo com fundo
branco}
        Canvas.Pen.Color:=clBlue;
        Canvas.Brush.Color:=clNone;
  end;
end;

{=============================================================================
 =============================================================================
                          OPERAÇÕES NO GRAFO
 =============================================================================
 =============================================================================}

procedure TFrmMapa.AplDadosGrafo(var PAux:PTRGrafo;RegGrafo:Grafo);
begin
  {Atribui os valores do RegGrafo ao ponteiro para essa estrutura (PAux)}
  PAux^.CodNodo:=RegGrafo.CodNodo;
  PAux^.nome_nodo:=RegGrafo.nome_nodo;
  PAux^.Opcao_tipo:=RegGrafo.opcao_tipo;
  PAux^.objetivo:=RegGrafo.objetivo;
  PAux^.pergunta:=RegGrafo.pergunta;
  PAux^.motivo:=RegGrafo.motivo;
  PAux^.relacionamento1:=RegGrafo.relacionamento1;
  PAux^.nome_valor1:=RegGrafo.nome_valor1;
  PAux^.relacionamento2:=RegGrafo.relacionamento2;
  PAux^.nome_valor2:=RegGrafo.nome_valor2;
  PAux^.xiNodo:=RegGrafo.xiNodo;
  PAux^.yiNodo:=RegGrafo.yiNodo;
  PAux^.xfNodo:=RegGrafo.xfNodo;
  PAux^.yfNodo:=RegGrafo.yfNodo;
  PAux^.Prox:=RegGrafo.Prox;
  PAux^.PrimAdj:=RegGrafo.PrimAdj;
  PAux^.UltAdj:=RegGrafo.UltAdj;
end;

  {Procedimento IncluiGrafo: Inclui um nodo no grafo.
 Parâmetros: PrimGrafo - Aponta para o primeiro elemento do gráfico
             UltGrafo - Aponta para o último elemento do grafo
             RegGrafo - Estrutura contendo os dados que serão incluídos}

procedure TFrmMapa.IncluiGrafo(var PrimGrafo:PTRGrafo;
          var UltGrafo:PTRGrafo;RegGrafo:Grafo);
var PAux:PTRGrafo;
begin
  new(PAux); {Aloca o espaço necessário}
  AplDadosGrafo(PAux,RegGrafo); {Atribui os dados do Nodo RegGrafo ao
ponteiro PAux}
  if PrimGrafo=nil then begin{Insere o primeiro nó do Grafo}
     PrimGrafo:=PAux;
     UltGrafo:=PAux;
     end
  else begin {Insere um nó depois do último da Lista}
     UltGrafo^.Prox:=PAux;
     UltGrafo:=PAux;
  end;
  PAtualGrafo:=PAux; {Ponteiro Atual recebe esse novo nó}
end;
```

```
{Recebe uma string como parâmetro e consulta ela no grafo.A chave de pes-
quisa do grafo é o campo Nome_nodo}
procedure TFrmMapa.ConsultaGrafo(PrimGrafo:PTRGrafo;Chave:str100);
var
  PAux:PTRGrafo;
  Achou:Boolean;
begin
  PAux:=PrimGrafo;
  Achou:=False;
  while(PAux<>nil) and (not Achou) do begin
       if PAux^.nome_nodo=Chave then Achou:=True
       else PAux:=PAux^.Prox;
  end;
  if Achou then begin
     PAtualGrafo:=PAux; {Ponteiro do Atual recebe o ponteiro encontrado}
     AtuJanNodo; {Atualiza a janela de dados sobre o Nodo}
  end;
end;

{Recebe o código de um nodo e realiza a pesquisa desse código no grafo,
se encontrar retorna um ponteiro para o endereço onde essa variável está
armazenada.}
function TFrmMa-
pa.AtribPtrGrafoCod(PrimGrafo:PTRGrafo;Chave:str2):PTRGrafo;
var PAux:PTRGrafo;
begin
  PAux:=PrimGrafo;
  while(PAux<>nil) do begin
       if PAux^.CodNodo=Chave then break
       else PAux:=PAux^.Prox;
  end;
  AtribPtrGrafoCod:=PAux;
end;

{=========================================================================
 =========================================================================
                        OPERAÇOES NOS NODOS
 =========================================================================
 =========================================================================}

 {Função que verifica se o código passado por parâmetro está no grafo}
function TFrmMapa.InGrafo(PrimGrafo:PTRGrafo;CodNodo:str2):boolean;
var PAux:PTRGrafo;
    Achou:Boolean;
begin
  PAux:=PrimGrafo;
  Achou:=False;
  while(PAux<>nil) and (not Achou) do begin
       if CodNodo=PAux^.CodNodo then Achou:=True
       else PAux:=PAux^.Prox;
  end;
  if Achou then InGrafo:=True {Retorna True se esse código existe}
  else InGrafo:=False;
end;

   {Atualiza a janela de dados sobre o nodo}
procedure TFrmMapa.AtuJanNodo;
begin
   if PAtualGrafo<>nil then begin {verifica se o PAtualGrafo está apon-
```

```
tando para alguém Atualiza os objetos com os valores do PAtualGrafo}
     {Para Atualizar RGTipo}
     If PAtualGrafo^.Opcao_tipo = numerica then
     RgTipo.ItemIndex := 1
     else
         if PAtualGrafo^.Opcao_tipo = Univalorada then
            RgTipo.ItemIndex := 0
                       else
                           if PAtualGrafo^.Opcao_tipo = multivalorada
then
                              RgTipo.ItemIndex := 2;
     {Para Atualizar RGObjetivo}
     If PAtualGrafo^.Objetivo = true then
             RGObjetivo.ItemIndex := 0
             else
                RGObjetivo.ItemIndex := 1;
     { Atualizar os valores}
     EdCodNodo.Text:=PAtualGrafo^.CodNodo;
     EdNomeNodo.Text:=PAtualGrafo^.nome_nodo;
     EdPergunta.Text := PAtualGrafo^.Pergunta;
     EdMotivo.Text := PAtualGrafo^.Motivo;
     CmbGrafo.Text:=PAtualGrafo^.nome_nodo;
   end
    else begin {Se não tiver ninguém no grafo, limpa tudo e coloca o de-
fault para os botões de rádio tipo de variável e objetivo}
     EdCodNodo.Clear;
     EdNomeNodo.Clear;
     EdPergunta.Clear;
     EdMotivo.Clear;
     CmbGrafo.Clear;
     RGTipo.ItemIndex:=0;
     RGObjetivo.ItemIndex:=1;
    end;
end;

{Função que recebe uma chave de pesquisa para o nome do nodo e retorna um
ponteiro para o endereço onde estão localizado os dados sobre esse nodo}
function TFrmMa-
pa.AtribPtrGrafo(PrimGrafo:PTRGrafo;Chave:str100):PTRGrafo;
var PAux:PTRGrafo;
begin
  PAux:=PrimGrafo;
  while(PAux<>nil) do begin
       if PAux^.nome_nodo=Chave then break
       else PAux:=PAux^.Prox;
  end;
  AtribPtrGrafo:=PAux; {Retorna um ponteiro para PAux}
                       {Observe que, se não existir nenhum nó com esse
                        nome, PAux irá retornar NULL}
end;

{Exclui de todas as listas adjacentes do grafo o elemento apontado por
 PAtualGrafo.
 Utilizado quando esse elemento é excluído do grafo}
procedure TFrmMapa.ExcluiLigacoes(PAtualGrafo:PTRGrafo);
var PAuxGrafo:PTRGrafo;
    PAuxAdj:PTRListaAdj;
begin
  PAuxGrafo:=PrimGrafo;
```

```
  while(PAuxGrafo<>nil) do begin {começa a pesquisa no grafo}
       PAuxAdj:=PAuxGrafo^.PrimAdj;
       while(PAuxAdj<>nil) do begin {começa a pesquisa na lista de adja-
centes}
            if PAuxAdj^.Nodo=PAtualGrafo then begin {Se o PAtualGrafo
estiver na lista -->}
{Exclui PAtual da Lista}
               ExcluiCami-
nho(PAuxGrafo^.PrimAdj,PAuxGrafo^.UltAdj,PAuxAdj);
               break;
               end
            else PAuxAdj:=PAuxAdj^.Prox;
       end;
       PAuxGrafo:=PAuxGrafo^.Prox;
  end;
end;

{=========================================================================
 =========================================================================
                      OPERAÇOES SOBRE OS ARCOS - VALORES
 =========================================================================
 =========================================================================}

{Inclui/Altera elemento (PDestino) na lista de adjacência do nodo
 identificado por POrigem}
procedure TFrmMa-
pa.AtribDistGrafo(POrigem,PDestino:PTRGrafo;relacionamento1:str2;
          nome_valor1:str100;relacionamento2:str2;nome_valor2:str100);
var PAuxAdj,PAntAdj:PTRListaAdj;
    Existe:Boolean;
begin
  PAuxAdj:=POrigem^.PrimAdj; {PAuxAdj recebe o primeiro da lista adjacen-
te de POrigem}
  Existe:=False; {Variável utilizada para identificar se é um novo ele-
mento ou não}
  if PAuxAdj=nil then begin{Inclui o primeiro da lista de adjacência}
     new(PAuxAdj);
     PAuxAdj^.Nodo:=PDestino;
     PAuxAdj^.Relacionamento1 := Relacionamento1;
     PAuxAdj^.nome_valor1:=nome_valor1;
     PAuxAdj^.Relacionamento2 := Relacionamento2;
     PAuxAdj^.nome_valor2:=nome_valor2;
     PAuxAdj^.Prox:=nil;
     PAuxAdj^.Ant:=nil;
     POrigem^.PrimAdj:=PAuxAdj;
     POrigem^.UltAdj:=PAuxAdj;
     Existe:=True;
     end
  else begin
     while(PAuxAdj<>nil) do begin
          if PAuxAdj^.Nodo=PDestino then begin {Se encontrar esse endere-
ço na lista de adjacência}
             PAuxAdj^.Relacionamento1 := Relacionamento1;{Altera o rela-
cionamento para o novo valor}
             PAuxAdj^.nome_valor1:=nome_valor1;    {Altera o valor do
nodo}
             PAuxAdj^.Relacionamento2 := Relacionamento2;{Altera o rela-
cionamento para o novo valor}
             PAuxAdj^.nome_valor2:=nome_valor2;    {Altera o valor do
```

```
nodo}
              Existe:=True;
              break;
              end
           else begin
                PAntAdj:=PAuxAdj;
                PAuxAdj:=PAuxAdj^.Prox;
                end
     end;
  end;
  if (not Existe) then begin {Se não existir PDestino na lista de adja-
cência então:}
     new(PAuxAdj);           {Aloca um espaço e insere esse novo elemen-
to}
     PAuxAdj^.Nodo:=PDestino;
     PAuxAdj^.Relacionamento1 := Relacionamento1;
     PAuxAdj^.nome_valor1:=nome_valor1;
     PAuxAdj^.Relacionamento2 := Relacionamento2;
     PAuxAdj^.nome_valor2:=nome_valor2;
     PAuxAdj^.Prox:=nil;
     PAuxAdj^.Ant:=PAntAdj;
     POrigem^.UltAdj^.Prox:=PAuxAdj;
     POrigem^.UltAdj:=PAuxAdj;
  end;
end;

{Atribui ao objeto cmbRelacionamento1 e cmbRelacionamento os sinais de
atribuição e a EdValorVar1 e EdValor2 os valores definidos entre dois
nodos selecionados apontadas por POrigem e PDestino}
procedure TFrmMapa.ExibeValor_Nodo(PrimGrafo,POrigem,PDestino:PTRGrafo);
var PAuxGrafo:PTRGrafo;
    PAuxAdj:PTRListaAdj;
    Achou:Boolean;
begin
  PAuxGrafo:=PrimGrafo;
  Achou:=False;
  EdValor1.Text:='';
  cmbRelacionamento1.ItemIndex := 0;
  EdValor2.Text:='';
  cmbRelacionamento2.ItemIndex := 0;

    while(PAuxGrafo<>nil) and (not Achou) do begin {Começa a pesquisa na
lista principal do grafo}
       if PAuxGrafo=POrigem then begin
          PAuxAdj:=PAuxGrafo^.PrimAdj; {Começa a pesquisa na lista de
adjacência}
          while(PAuxAdj<>nil) do begin
               if PAuxAdj^.Nodo=PDestino then begin {Se os endereços de
Paux e PDestino forem iguais --> achou}
                  {Atualiza o objeto}
                  CmbRelacionamento1.text := PAuxAdj^.relacionamento1;
                  EdValor1.Text:=PAuxAdj^.nome_valor1;
                  CmbRelacionamento2.text := PAuxAdj^.relacionamento2;
                  EdValor2.Text:=PAuxAdj^.nome_valor2;
                  if (CmbRelacionamento2.Text <> '') or (EdValor2.Text <>
'') then begin
                  CmbRelacionamento2.Visible := true;
                  EdValor2.Visible :=true;
                  end
```

```
                    else begin
                         CmbRelacionamento2.Visible := false;
                         EdValor2.Visible :=false;
                    end;
                  RadioSim.Checked:=False;
                  RadioNao.Checked:=True;
                  Achou:=True;
                  break;
                  end
               else PAuxAdj:=PAuxAdj^.Prox;
          end;
          PAuxGrafo:=PAuxGrafo^.Prox;
          end
       else PAuxGrafo:=PAuxGrafo^.Prox;
  end;
end;

{Exclusão de caminhos que em conjunto formarão as premissas e a cauda das
regras - exclusão na lista de adjacentes}
procedure TFrmMapa.ExcluiCaminho(var PrimAdj:PTRListaAdj;var Ul-
tAdj:PTRListaAdj;
                                 var PAuxAdj:PTRListaAdj);
begin
  if PAuxAdj=PrimAdj then begin {Exclui o primeiro da lista de adjacen-
tes}
     PrimAdj:=PrimAdj^.Prox;
     if PrimAdj<>nil then PrimAdj^.Ant:=nil;
     PAuxAdj^.Prox:=nil;
     Dispose(PAuxAdj);
     end
  else if PAuxAdj=UltAdj then begin {Exclui o último da lista de adjacen-
tes}
     UltAdj:=UltAdj^.Ant;
     UltAdj^.Prox:=nil;
     Dispose(PAuxAdj);
     end
  else begin {Exclui um elemento que está entre dois}
     PAuxAdj^.Ant^.Prox:=PAuxAdj^.Prox;
     PAuxAdj^.Prox^.Ant:=PAuxAdj^.Ant;  {É duplamente encadeada!!!}
     PAuxAdj^.Prox:=nil;
     Dispose(PAuxAdj);
  end;
end;

{=========================================================================
 =========================================================================
                      LISTA DE ORIGEM E LISTA DE DESTINO
 =========================================================================
 =========================================================================}

 {Utilizada para saber se um nó esta na lista de todos os adjacentes}
Function TfrmMapa.EstaNosAdjacentes(var Nodo:PTRGrafo):boolean;
var PAuxGrafo:PTRGrafo;
    PAuxAdj:PTRListaAdj;
    Encontrado:boolean;
begin
  Encontrado:=false;
  PAuxGrafo:=PrimGrafo; {Para iniciar no Começo do Grafo};
  while(PAuxGrafo<>nil) and (not encontrado) do begin {começa a pesquisa
```

```
no grafo}
       PAuxAdj:=PAuxGrafo^.PrimAdj;
       while(PAuxAdj<>nil) do begin {começa a pesquisa na lista de adja-
centes}
            if PAuxAdj^.Nodo=PAuxGrafo then begin
               Encontrado:=true;
               break;
               end
            else PAuxAdj:=PAuxAdj^.Prox;
       end;
  PAuxGrafo:=PAuxGrafo^.Prox;
  end;
  EstaNosAdjacentes:=Encontrado;
end;

{Procedimento para formar a lista de Origem}
Procedure TFrmMapa.FormarListaOrigem;
var PAux:PTRGrafo;
Begin
PAux:=PrimGrafo; {Para iniciar no Começo do Grafo};
while PAux<> nil do begin
  if not EstaNosAdjacentes(PAux) then
  IncListaOrigem(PrimListaOrigem,UltListaOrigem,PAux);
  PAux:=Paux.Prox;
 end;
end;

{Procedimento para incluir um nodo na lista de Origem}
 procedure TFrmMapa.IncListaOrigem(var PrimListaOrigem:PTRListaOrigem;
          var UltListaOrigem:PTRListaOrigem;Nodo:PTRGrafo);
var PAux:PTRListaOrigem;
begin
  new(PAux);
  PAux^.NodoOrigem :=Nodo;
  PAux^.Prox:=nil;
  if PrimListaOrigem=nil then begin {Se for o primeiro...}
     PrimListaOrigem:=PAux;
     UltListaOrigem:=PAux;
     end
  else begin {Senão, inclui depois do último}
     UltListaOrigem^.Prox :=PAux;
     UltListaOrigem:=PAux;
  end;
end;

{Procedimento para formar a lista de Destino}
Procedure TFrmMapa.FormarListaDestino;
var PAux:PTRGrafo;
Begin
PAux:=PrimGrafo; {Para iniciar no Começo do Grafo};
while PAux<> nil do begin
 if (PAux.Objetivo=true) then
  IncListaDestino(PrimListaDestino,UltListaDestino,PAux);
  PAux:=PAux.Prox;
 end;
end;

{Procedimento par incluir um nodo na lista de Destino}
procedure TFrmMapa.IncListaDestino(var PrimListaDestino:PTRListaDestino;
```

```
          var UltListaDestino:PTRListaDestino;Nodo:PTRGrafo);
var PAux:PTRListaDestino;
begin
  new(PAux);
  PAux^.NodoDestino:=Nodo;
  PAux^.Prox:=nil;
  if PrimListaDestino=nil then begin {Se for o primeiro...}
     PrimListaDestino:=PAux;
     UltListaDestino:=PAux;
     end
  else begin {Senão, inclui depois do último}
     UltListaDestino^.Prox :=PAux;
     UltListaDestino:=PAux;
  end;
end;

{Limpa o conteúdo da Lista Encadeada que armazena os nodos fonte (ListaO-
rigem) É executado quando um novo caminhamento é feito, ou quando o pro-
grama é terminado}
procedure TFrmMapa.LimpaListaOrigem(var PrimListaOrigem:PTRListaOrigem;
          var UltListaOrigem:PTRListaOrigem);
var PAux:PTRListaOrigem;
begin
  PAux:=PrimListaOrigem;
  while(PAux<>nil) do begin {Enquanto existir elementos na lista}
       PrimListaOrigem:=PAux^.Prox;
       PAux^.Prox:=nil;
       Dispose(PAux);  {Desaloca}
       PAux:=PrimListaOrigem;
  end;
  UltListaOrigem:=PrimListaOrigem; {null}
  DesMapa;
end;

{Limpa o conteúdo da Lista Encadeada que armazena os nodos destino (Lis-
taDestino). É executado quando um novo caminhamento é feito, ou quando o
programa é terminado}
procedure TfrmMapa.LimpaListaDestino(var PrimListaDesti-
no:PTRListaDestino;
          var UltListaDestino:PTRListaDestino);
var PAux:PTRListaDestino;
begin
  PAux:=PrimListaDestino;
  while(PAux<>nil) do begin {Enquanto existir elementos na lista}
       PrimListaDestino:=PAux^.Prox;
       PAux^.Prox:=nil;
       Dispose(PAux);  {Desaloca}
       PAux:=PrimListaDestino;
  end;
  UltListaDestino:=PrimListaDestino; {null}
  DesMapa;
end;

{=============================================================================
 =============================================================================
                        REGRAS
 =============================================================================
 =============================================================================}
```

```
   {Procedimento que realiza o caminhamento entre dois nodos do grafo
 Recebe o nodo origem, o nodo destino e a estrutura que receberá os re-
sultados da formação de regras.
 Caminhamento por PROFUNDIDADE - Estrutura auxiliar: Pilha}
procedure TFrmMapa.ConstruirRegras(POrigem,PDestino:PTRGrafo);
var PAuxAdj:PTRListaAdj;
    begin
  if POrigem=PDestino then begin {Se chegou ao nodo}
     Empilha(POrigem); {Empilha a origem}
     AtuJanRegras; {Atualiza janela de Regras}
     Desempilha;
     end
  else begin
     Empilha(POrigem); {Empilha a origem}
     PAuxAdj:=POrigem^.PrimAdj; {começa a pesquisar os nodos adjacentes}
     while PAuxAdj<>nil do begin
          if not Visitada(PAuxAdj^.Nodo) then begin {se não estiver na
pilha}
                  Topo^.relacionamento1:=PAuxAdj^.relacionamento1; {A-
tribui o sinal ao valor do nodo}
                 Topo^.nome_valor1 :=PAuxAdj^.nome_valor1; {Atribui o
valor do nodo}
                  Topo^.relacionamento2:=PAuxAdj^.relacionamento2; {A-
tribui o sinal ao valor do nodo - Intervalo}
                 Topo^.nome_valor2 :=PAuxAdj^.nome_valor2; {Atribui o
valor do nodo - Intervalo}
                 ConstruirRegras(PAuxAdj^.Nodo,PDestino); {Executa for-
mação de regras com a nova Origem **Recursão}
          end;
            PAuxAdj:=PAuxAdj^.Prox; {Vai para o próximo da lista de adja-
centes}
     end;
     Desempilha;
  end;
end;

{Recebe o endereço de um elemento do grafo e retorna verdadeiro se ele
 estiver na pilha de regras.
 Essa crítica serve para que um mesmo nodo não seja visitado mais de
 uma vez a cada caminhamento}
function TFrmMapa.Visitada(Nodo:PTRGrafo):Boolean;
var PAux:PTRPilhaRegra;
    Achei:Boolean;
begin
  PAux:=Topo;
  Achei:=False;
  {Pesquisa em toda a pilha}
  while(PAux<>nil) and (not Achei) do begin
       if PAux^.Nodo=Nodo then Achei:=True
       else PAux:=PAux^.Prox;
  end;
  Visitada:=Achei;
end;

{Empilha um nodo na Pilha de Regras}
procedure TFrmMapa.Empilha(Nodo:PTRGrafo);
var PAux:PTRPilhaRegra;
begin
  new(PAux);
```

```
  PAux^.Nodo:=Nodo;
  PAux^.relacionamento1:='';
  PAux^.nome_valor1:='';
  PAux^.relacionamento2:='';
  PAux^.nome_valor2:='';
  PAux^.Prox:=Topo;
  Topo:=PAux;
end;

{Desempilha um nodo da Pilha de Regras}
procedure TFrmMapa.Desempilha;
var PAux:PTRPilhaRegra;
begin
  PAux:=Topo;
  if Topo<>nil then begin
     Topo:=Topo^.Prox;
     Dispose(PAux);
  end;
end;

{Procedimento que inclui um novo nodo na lista encadeada ListaRegras}
procedure TFrmMapa.IncListaRegra(var PrimListaRegras:PTRListaRegra;
          var UltListaRegra:PTRListaRegra;linha:longint;Nodo:PTRGrafo);
var PAux:PTRListaRegra;
begin
  new(PAux);
  PAux^.Nodo:=Nodo;
  PAux^.Linha:=Linha;
  PAux^.Prox:=nil;
  if PrimListaRegras=nil then begin {Se for o primeiro...}
     PrimListaRegras:=PAux;
     UltListaRegra:=PAux;
     end
  else begin {Senão, inclui depois do último}
     UltListaRegra^.Prox:=PAux;
     UltListaRegra:=PAux;
  end;
end;

{Finalmente o Caminhamento entre dois nodos
 Procedimento para formação das Regras}
procedure TFrmMapa.FormaRegra;
var PTROrigem,PTRDestino:PTRGrafo;
    PAuxListaOrigem:PTRListaOrigem;
    PAuxListaDestino:PTRListaDestino;
begin
   {Limpa Lista de regras, lista de origem e de destino};
  LimpaListaRegra(PrimListaRegra,UltListaRegra);
  LimpaListaOrigem(PrimListaOrigem,UltListaOrigem);
  LimpaListaDestino(PrimListaDestino,UltListadestino);;
  FrmResultado.RERelatorio.Lines.Add('');
  FrmResultado.RERelatorio.Lines.Add('');
  FrmResultado.RERelatorio.Lines.Add('REGRAS');
  FrmResultado.RERelatorio.Lines.Add('');
  FrmResultado.RERelatorio.Lines.Add('');
  {Atualiza ponteiros para os nodos de origem e destino}
  FormarListaOrigem;
  FormarListaDestino;
  if PrimListaDestino = nil then begin
```

```
     showmessage('O Grafo não apresenta objetivos definidos');
     exit;
  end;
  PAuxListaOrigem:=PrimListaOrigem;
    while PAuxListaOrigem  <> nil do begin
      PTROrigem:=PAuxListaOrigem^.NodoOrigem;
      PAuxListaDestino:=PrimListaDestino;
      while PAuxListaDestino <> nil do begin
          PTRDestino:=PAuxListaDestino^.NodoDestino;
          if (PTROrigem=nil) or (PTRDestino=nil) then  exit;
          ConstruirRegras(PTROrigem,PTRDestino); {Executa o procedimento
recursivo forma regra}
      PAuxListaDestino:=PAuxListaDestino.Prox;
      end;
  PAuxListaOrigem:=PAuxListaOrigem.Prox;
  end;
  end;

{=============================================================================
 =============================================================================
                      VARIÁVEL DO SISTEMA ESPECIALISTA
 =============================================================================
 =============================================================================}

 {Procedimento que atribui um nome e define o tipo da variável objetivo}
procedure TFrmMapa.definicaoObjetivo;
begin
{Para receber o nome da variável objetivo}
Resultado:=InputBox('Nome da Variável objetivo','Entre com o nome da va-
riável Objetivo',
            'Resultado');
{Para receber o tipo da variável objetivo}
TipoObjetivo:=InputBox('Tipo da Variável objetivo',
            'Tipo da Variável Objetivo(Univalorada-Multivalorada)'
            ,'Univalorada');
    while ((TipoObjetivo <> 'Univalorada') and (TipoObjetivo <> 'Multiva-
lorada') and
        (TipoObjetivo <> 'univalorada') and (TipoObjetivo <> 'multivalo-
rada')) do begin
            ShowMessage('Entre com um valor válido');
             TipoObjetivo:=InputBox('Tipo da Variável objetivo',
             'Tipo da Variável Objetivo(Univalorada-Multivalorada)'
             ,'Univalorada');
    end;
 end;

{Procedimento para relacionar as variáveis e seus respectivos valores que
irão formar o Sistema Especialista}
procedure TFrmMapa.VariaveisDoSistema;
var PAuxGrafo:PTRGrafo;
    PAuxAdj:PTRListaAdj;
    Opcao:string;
begin
PAuxGrafo:=PrimGrafo;
{Variáveis que não são objetivos do Sistema}
FrmResultado.RERelatorio.Lines.Add('VARIÁVEIS');
FrmResultado.RERelatorio.Lines.Add('');
while PAuxGrafo <> nil do begin
  if PAuxgrafo.Objetivo = false then begin
```

```
       {Para receber o valor do tipo de variável - transformação de um
tipo enumerado em uma string}
        If PAuxGrafo.Opcao_tipo = numerica then
           opcao:='numérica'
           else if PAuxGrafo.Opcao_tipo = univalorada
            then opcao:= 'univalorada'
              else opcao:= 'multivalorada';
       FrmResultado.RERelatorio.Lines.Add('     '+'NOME:
'+PAuxGrafo.Nome_nodo);
       FrmResultado.RERelatorio.Lines.Add('        '+'OBJETIVO: NÃO');
       FrmResultado.RERelatorio.Lines.Add('        '+'TIPO: '+ opcao);
       FrmResultado.RERelatorio.Lines.Add('        '+'PERGUNTA:
'+PAuxGrafo.Pergunta);
       FrmResultado.RERelatorio.Lines.Add('        '+'MOTIVO DA PERGUNTA:
'+
                                                  PAuxGrafo.Motivo);
       FrmResultado.RERelatorio.Lines.Add('        '+ 'VALORES');
       PAuxAdj:=PAuxGrafo^.PrimAdj;
        {Para encontrar os Valores de Cada Variável Visitamos os nodos
adjacentes}
            while(PAuxAdj<>nil) do begin {começa a pesquisa na lista de
adjacentes}
            FrmResultado.RERelatorio.Lines.Add('
'+PAuxAdj.Relacionamento1+
               ' '+PAuxAdj.nome_valor1+'
'+PAuxAdj.Relacionamento2+' '+
               PAuxAdj.nome_valor2);
            PAuxAdj := PAuxAdj^.Prox;
             end;
    FrmResultado.RERelatorio.Lines.Add('');
  end;
  PAuxGrafo:=PAuxGrafo.Prox;
end;
  {Variáveis que são objetivos do Sistema}
  FrmResultado.RERelatorio.Lines.Add('');
  FrmResultado.RERelatorio.Lines.Add('     '+'NOME: '+ Resultado);
  FrmResultado.RERelatorio.Lines.Add('        '+'OBJETIVO: SIM');
  FrmResultado.RERelatorio.Lines.Add('        '+'Tipo: '+ TipoObjetivo);
  FrmResultado.RERelatorio.Lines.Add('        '+ 'VALORES');
  PAuxGrafo:=PrimGrafo;
  {Os Valores são os nodos marcados com Objetivo}
  While PAuxGrafo<> nil do begin
        if PAuxGrafo.Objetivo = true then
              FrmResultado.RERelatorio.Lines.Add('           '+'=
'+PAuxGrafo.Nome_nodo);
        PAuxGrafo:=PAuxGrafo.Prox;
  end;
end;
{=============================================================================
 =============================================================================
                      OPERAÇÕES NOS BOTÕES DE RÁDIO
 =============================================================================
 =============================================================================}

 {Quando o nodo é objetivo já foi definido seu nome e seu tipo - O seu
nome será portanto um valor}
procedure TFrmMapa.RGObjetivoClick(Sender: TObject);
begin
     If RGObjetivo.ItemIndex = 0 then
```

```
        begin
        RGTipo.Visible:= false;
        labelPergunta.Visible := false;
        EdPergunta.Visible := false;
        labelMotivo.Visible := false;
        EdMotivo.Visible := false;
        end
            else
            if RGObjetivo.ItemIndex = 1 then
            begin
               RgTipo.Visible:=true;
               labelPergunta.Visible := true;
               EdPergunta.Visible := true;
               labelMotivo.Visible := true;
               EdMotivo.Visible := true;
            end;
        end;

{ Executado para definição se o valor de um arco é um número ou um inter-
valo}
procedure TFrmMapa.RGIntervaloClick(Sender: TObject);
begin
     if RGIntervalo.ItemIndex = 0 then begin
        CmbRelacionamento1.Items.Clear;
        CmbRelacionamento2.Items.Clear;
        CmbRelacionamento2.Visible := true;
        EdValor2.Visible := true;
        CmbRelacionamento1.Items.Add('>');
        CmbRelacionamento1.Items.Add('>=');
        CmbRelacionamento1.ItemIndex:=0;
        CmbRelacionamento2.Items.Add('');
        CmbRelacionamento2.Items.Add('<');
        CmbRelacionamento2.Items.Add('<=');
        CmbRelacionamento2.ItemIndex:=0;
      end
     else  begin
        CmbRelacionamento1.Items.Clear;
        CmbRelacionamento2.Items.Clear;
        CmbRelacionamento2.Visible := false;
        EdValor2.Visible := false;
        CmbRelacionamento1.Items.Add('=');
        CmbRelacionamento1.Items.Add('<>');
        CmbRelacionamento1.ItemIndex := 0;
     end;
end;

 {=========================================================================
 =========================================================================
                      OPERAÇÕES NOS COMPONENTES CAMBOS
 =========================================================================
 =========================================================================}

{Procedimento AtualizaCmbGrafo. Atualiza os objetos CmbGrafo, CmbOrigem,
CmbDestino, CmbOrigemCam, CmbDestinoCam com os nodos atuais do grafo}
procedure TFrmMapa.AtualizaCmbGrafo;
var PAux:PTRGrafo;
begin
  PAux:=PrimGrafo;
  CmbGrafo.Clear; {Limpa o conteúdo dos objetos}
```

```
  CmbOrigem.Clear;
  CmbDestino.Clear;
  while(PAux<>nil) do begin
       {Adiciona a cada objeto um novo item que é o nome do nodo}
       CmbGrafo.Items.Add(PAux^.nome_nodo);
       CmbOrigem.Items.Add(PAux^.nome_nodo);
       CmbDestino.Items.Add(PAux^.nome_nodo);
       PAux:=PAux^.Prox;
  end;
end;

{Executado quando o objeto CmbGrafo é clicado}
procedure TFrmMapa.CmbGrafoClick(Sender: TObject);
var Chave:str100;
begin
  Chave:=CmbGrafo.Items[CmbGrafo.ItemIndex]; {Chave recebe o nome do nodo
clicado}
  ConsultaGrafo(PrimGrafo,Chave); {Consulta essa chave no Grafo}
  EdCodNodo.SetFocus; {Volta o foco para o objeto EdCodNodo}
end;

{Executado quando é alterado o nodo no objeto CmbOrigem}
procedure TFrmMapa.CmbOrigemChange(Sender: TObject);
var POrigem,PDestino:PTRGrafo;
begin
 CmbRelacionamento1.Clear;
 CmbRelacionamento1.Items.Add('=');
 CmbRelacionamento1.Items.Add('<>');
 CmbRelacionamento1.ItemIndex:=0;
 RGIntervalo.Visible:=false;
  {POrigem e PDestino apontam para o endereço do grafo onde está locali-
zada o nodo selecionado no objeto}
  POrigem:=AtribPTRGrafo(PrimGrafo,CmbOrigem.Items[CmbOrigem.ItemIndex]);
  {Retorna um ponteiro para o grafo}
      { Atualiza o relacionamento de atribuição de valor para um nodo do
tipo numérico}
    if POrigem^.Opcao_tipo = numerica then begin
      RgIntervalo.Visible :=true;
      RgIntervalo.ItemIndex :=1;
      end
    { Ou não numérico}
   else
    if POrigem^.Opcao_tipo <> numerica then begin
      RgIntervalo.Visible:=false;
      CmbRelacionamento2.Visible :=false;
      EdValor2.Visible :=false;
    end;
   PDesti-
no:=AtribPTRGrafo(PrimGrafo,CmbDestino.Items[CmbDestino.ItemIndex]);
  {Retorna um ponteiro para o grafo}
  ExibeValor_Nodo(PrimGrafo,POrigem,PDestino); {Procedimento que mostra o
valor entre POrigem e PDestino}
  {Não Permite que um nodo objetivo dê origem a outro nodo - O nodo obje-
tivo é do tipo sumidouro}
  if POrigem^.Objetivo = true then begin
     ShowMessage('Um nodo tipo objetivo só pode ser destino');
     CmbOrigem.ItemIndex := -1;
     exit;
  end;
```

```
end;

{Executado quando é alterado o nodo no objeto CmbOrigem. Igual ao
TForm1.CmbOrigemChange}
procedure TFrmMapa.CmbDestinoChange(Sender: TObject);
var POrigem,PDestino:PTRGrafo;
begin
  if (CmbOrigem.ItemIndex=-1) then exit;
  POrigem:=AtribPTRGrafo(PrimGrafo,CmbOrigem.Items[CmbOrigem.ItemIndex]);
  PDesti-
no:=AtribPTRGrafo(PrimGrafo,CmbDestino.Items[CmbDestino.ItemIndex]);
  ExibeValor_Nodo(PrimGrafo,POrigem,PDestino);
end;

{Executado quando o objeto Nodo do objeto origem é modificada}
procedure TFrmMapa.CmbOrigemCamChange(Sender: TObject);
begin
  {Limpa a lista de regras, para que possa ser feito um novo caminhamen-
to}
  LimpaListaRegra(PrimListaRegra,UltListaRegra);
end;

{Executado quando o objeto Nodo do objeto origem é modificada}
procedure TFrmMapa.CmbDestinoCamChange(Sender: TObject);
begin
  {Limpa a lista de regras, para que possa ser feito um novo caminhamen-
to}
  LimpaListaregra(PrimListaRegra,UltListaRegra);
end;

{=========================================================================
 =========================================================================
                        COMPONENTES - BOTÕES
 =========================================================================
 =========================================================================}

{Executado quando o Botão Inclui é clicado}
procedure TFrmMapa.BtnIncluiClick(Sender: TObject);
var Opcao_tipo:Tipovariavel;
    objetivo:boolean;
begin
  if (EdCodNodo.Text='') or (EdCodNodo.Text='-') then begin
     {Não deixa que o código do Nodo receba o caractere '-' pois ele é
usado na formatação do arquivo de saída}
     ShowMessage('O Campo código está vazio ou preenchido incorretamen-
te.');
     EdCodNodo.SetFocus ;
     exit
     end
  else if InGrafo(PrimGrafo,EdCodNodo.Text) then begin
     {Se já existir esse código}
     ShowMessage('Esse código já existe.');
     exit;
  end;
    {Recebe o valor para Tipo de variável - Numérica, Univalorada, Multi-
valorada}
  If RgTipo.ItemIndex = 1 then
     Opcao_tipo := numerica
     else
```

```
          if RgTipo.ItemIndex = 0 then
             Opcao_tipo:= Univalorada
                        else
                            if RgTipo.ItemIndex = 2 then
                               Opcao_tipo:= multivalorada;
  {Recebe o valor para Objetivo}
   if RgObjetivo.ItemIndex = 0 then
      objetivo := true
                else objetivo :=false;
   {Não deixa que o nodo fique sem um nome}
   while EdNomeNodo.Text='' do begin
       ShowMessage('O Campo Nome da Variável deve estar preenchido');
       EdNomeNodo.SetFocus;
       exit;
   end;
    if RGObjetivo.ItemIndex = 1 then begin
       while EdPergunta.Text='' do begin
           ShowMessage('É necessário uma pergunta para que a Variável
seja instanciada no SE');
           EdPergunta.SetFocus;
           exit;
      end;
   end;
  {Atualiza a variável RegGrafo com os valores do formulário}
  RegGrafo.CodNodo:=EdCodNodo.Text;
  RegGrafo.nome_nodo:=EdNomeNodo.Text;
  RegGrafo.Opcao_tipo := opcao_tipo;
  RegGrafo.Objetivo := objetivo;
  RegGrafo.Pergunta := EdPergunta.Text;
  RegGrafo.Motivo := EdMotivo.Text;
  RegGrafo.relacionamento1 := '';
  RegGrafo.nome_valor1:='';
  RegGrafo.relacionamento2 := '';
  RegGrafo.nome_valor2:='';
  RegGrafo.xiNodo:=10;
  RegGrafo.yiNodo:=10;
  RegGrafo.xfNodo:=10+TAMNODO;
  RegGrafo.yfNodo:=10+TAMNODO;
  RegGrafo.Prox:=nil;
  RegGrafo.PrimAdj:=nil;
  RegGrafo.UltAdj:=nil;
  {Inclusão no Grafo}
  IncluiGrafo(PrimGrafo,UltGrafo,RegGrafo);
  AtualizaCmbGrafo;
  {Desenho da Variável no Mapa}
  DesMapa;
  LimpaFormNodo;
  EdCodNodo.SetFocus;
end;

{Executado quando o botão Altera é Clicado}
procedure TFrmMapa.BtnAlteraClick(Sender: TObject);
var Opcao_tipo:Tipovariavel;
    objetivo:boolean;
begin
  if PrimGrafo=nil then begin
     ShowMessage('Não Há Nenhuma Variável Cadastrada');
     exit;
     end
```

```
  else if EdCodNodo.Text='' then begin
       ShowMessage('O Campo código da Variável deve estar preenchido');
       exit;
  end;
   If RgTipo.ItemIndex = 1 then
     Opcao_tipo := numerica
     else
         if RgTipo.ItemIndex = 0 then
            Opcao_tipo:= Univalorada
                       else
                           if RgTipo.ItemIndex = 2 then
                              Opcao_tipo:= multivalorada;
  {Recebe o valor para Objetivo}
   if RgObjetivo.ItemIndex = 0 then
      objetivo := true
                else objetivo :=false;
    {Atualiza os dados de PAtualGrafo}
  PAtualGrafo^.CodNodo:=EdCodNodo.Text;
  PAtualGrafo^.nome_nodo:=EdNomeNodo.Text;
  PAtualGrafo^.Pergunta :=EdPergunta.Text;
  PAtualGrafo^.Motivo := EdMotivo.Text ;
  PAtualGrafo^.Objetivo:= objetivo;
  PAtualGrafo^.Opcao_tipo := opcao_tipo;
  AtualizaCmbGrafo;
end;

{Executado quando o botão Exclui é clicado}
{Exclui do grafo o nodo que está sendo apontado por PAtualGrafo }
 procedure TFrmMapa.BtnExcluiClick(Sender: TObject);
var PAntGrafo,PAux:PTRGrafo;
begin
  if PrimGrafo=nil then begin
     ShowMessage('Não há Nenhuma Nodo Cadastrado');
     exit;
  end;
  PAux:=PrimGrafo;
  PAntGrafo:=PrimGrafo;
  {Primeiro Exclui todas as ligações dele}
  ExcluiLigacoes(PAtualGrafo);
  if PAtualGrafo=PrimGrafo then begin
     {Se for o primeiro do grafo}
     PrimGrafo:=PAtualGrafo^.Prox;
     PAtualGrafo^.Prox:=nil;
     Dispose(PAtualGrafo);
     PAntGrafo:=nil;
     end
  else while(PAux<>nil) do begin
       if PAtualGrafo=PAux then begin
          PAntGrafo^.Prox:=PAux^.Prox;{Próximo do anterior recebe próximo
do Paux}
          PAux^.Prox:=nil; {Próximo do PAux recebe nil}
          if PAux=UltGrafo then UltGrafo:=PAntGrafo; {Se o excluído for o
último, atualiza o último}
          Dispose(PAux);
          break;
          end
       else begin {Continua varrendo o grafo}
          PAntGrafo:=PAux;
          PAux:=PAux^.Prox;
```

```
        end;
  end;
  if PAntGrafo<>nil then PAtualGrafo:=PAntGrafo {Atual aponta para o an-
terior do excluído}
  else PAtualGrafo:=PrimGrafo; {Para não dar pau se o excluído for o pri-
meiro do grafo}
  AtualizaCmbGrafo;
  DesMapa;
  LimpaListaOrigem(PrimListaOrigem,UltListaOrigem);
  LimpaListaDestino(PrimListaDestino,UltListaDestino);
  LimpaListaRegra(PrimListaRegra,UltListaRegra); {Para não dar pau se a
variável excluída fazia parte de um dos caminhos}
  AtuJanNodo;
end;

{Executado quando o Botão Atribui é clicado
 Insere/Altera um valor entre dois nodos adjacentes}
procedure TFrmMapa.BtnAplDistClick(Sender: TObject);
var POrigem,PDestino:PTRGrafo;
begin
  while (CmbOrigem.Text='') or (CmbDestino.Text='') do begin
     ShowMessage('É necessário o preenchimento dos campos Origem e Desti-
no');
     exit;
  end;
  while (CmbOrigem.Text=CmbDestino.Text) do begin {Não permite o laço}
     ShowMessage('A variável origem é igual a variável destino');
     exit;
  end;
nome_valor1:=edValor1.Text;
relacionamento1 := CmbRelacionamento1.Text ;
nome_valor2:=edValor2.Text;
relacionamento2 := CmbRelacionamento2.Text ;
  {Pesquisa no grafo os nodo de origem e de destino, atribuindo os
   respectivos ponteiros}
  POrigem:=AtribPtrGrafo(PrimGrafo,CmbOrigem.Text); {Retorna um ponteiro
para POrigem}
  PDestino:=AtribPtrGrafo(PrimGrafo,CmbDestino.Text); {Retorna um pontei-
ro para PDestino}
   AtribDistGra-
fo(POrigem,PDestino,relacionamento1,nome_valor1,relacionamento2,nome_valo
r2); {Função de inclusão na lista de adjacência}
  if RadioSim.Checked=True then {Se for "mão-dupla" (não-dirigido) execu-
ta a mesma função trocando a origem pelo destino e o destino pela origem}
     AtribDistGra-
fo(PDestino,POrigem,relacionamento1,nome_valor1,relacionamento2,nome_valo
r2);
     DesMapa;
  { O preenchimento do campo valor1 é obrigatório}
  while EdValor1.Text = '' do begin
     ShowMessage('É obrigatório o preenchimento do valor');
     edValor1.SetFocus ;
     exit;
  end;
   {Se for permitido Intervalo é obrigatório o preenchimento da caixa de
texto2 com um valor numérico}
    if (RgIntervalo.ItemIndex = 0) then begin
       while (EdValor2.Text = '') do begin
          ShowMessage('É obrigatório o preenchimento do valor');
```

```
          edValor2.SetFocus ;
          exit;
       end;
  end;
 DesMapa;
 end;

{Executado quando o botão de exclusão de caminho é clicado
 Exclui uma ligação entre dois nodos}
procedure TFrmMapa.BtnExcClick(Sender: TObject);
var POrigem,PDestino:PTRGrafo;
    PAuxAdj:PTRListaAdj;
begin
  {Atribui os Ponteiros para o nodo origem e destino}
  POrigem:=AtribPTRGrafo(PrimGrafo,CmbOrigem.Items[CmbOrigem.ItemIndex]);
  PDesti-
no:=AtribPTRGrafo(PrimGrafo,CmbDestino.Items[CmbDestino.ItemIndex]);
  if (POrigem=nil) or (PDestino=nil) then exit;
  PAuxAdj:=POrigem^.PrimAdj;
  while(PAuxAdj<>nil) do begin {começa a pesquisa na lista de adjacência}
       if PAuxAdj^.Nodo=PDestino then begin {Se encontrar o PDestino na
lista de adjacência do POrigem}
          ExcluiCaminho(POrigem^.PrimAdj,POrigem^.UltAdj,PAuxAdj);
          break;
          end
       else PAuxAdj:=PAuxAdj^.Prox;
  end;
  LimpaListaOrigem(PrimListaOrigem,UltListaOrigem);
  LimpaListaDestino(PrimListaDestino,UltListaDestino);
  LimpaListaRegra(PrimListaRegra,UltListaRegra); {Para não dar pau se
excluir uma ligação que está presente na lista}
  DesMapa;
end;

{Executado quando o botão <Limpa> é clicado}
procedure TFrmMapa.BtnLimpaClick(Sender: TObject);
begin
  {Limpa os objetos}
  CmbGrafo.Text:='';
  EdCodNodo.Text:='';
  EdNomeNodo.Text:='';
  EdPergunta.Text :='';
  EdMotivo.Text :='';
  RGTipo.ItemIndex:=0;
  RGObjetivo.ItemIndex:=1;
  EdCodNodo.Setfocus;
end;

{Executado quando o botão de resultados é clicado}
procedure TFrmMapa.BtnResultadoClick(Sender: TObject);
begin
DefinicaoObjetivo;
FrmResultado.RERelatorio.Lines.Clear;
Numero:=1;{Atribui a 1ª regra o número 1}
  {Obtém todas as variáveis do Sistema Especialista}
  VariaveisDoSistema;
  {Obtém todas as Regras do sistema Especialista baseadas no Grafo}
  FrmMapa.FormaRegra ;
  {Exibe os Fatos e regras que alimentarão o Sistema Especialista}
```

```
  FrmResultado.ShowModal;
  DesMapa;
end;

{Executado quando o Botão sair é clicado}
procedure TFrmMapa.BitSaidaClick(Sender: TObject);
begin
 {Pergunta se não deseja salvar o arquivo}
  if MessageDlg('Deseja Salvar o Mapa em Arqui-
vo',mtConfirmation,[mbYes,mbNo],0)=idYes
     then begin
     SalvarDialog.FileName:=NomeArq;
     if SalvarDialog.Execute then begin
        NomeArq:=SalvarDialog.FileName;
        Salvar(NomeArq);
     end;
     end
  else begin
     LimpaGrafo(PrimGrafo,UltGrafo);
     LimpaListaOrigem(PrimListaOrigem,UltListaOrigem);
     LimpaListaDestino(PrimListaDestino,UltListaDestino);
     LimpaListaRegra(PrimListaRegra,UltListaRegra);
     DesMapa;
  end;
end;

{=========================================================================
 =========================================================================
                      COMPONENTES CAIXA DE TEXTO
 =========================================================================
 =========================================================================}

{Não permite o preenchimento com um valor que não seja um número, ',' ou
'.' para uma variável númerica na caixa de valor1}
procedure TFrmMapa.EdValor1KeyPress(Sender: TObject; var Key: Char);
 var POrigem:PTRGrafo;
begin
  POrigem:=AtribPTRGrafo(PrimGrafo,CmbOrigem.Items[CmbOrigem.ItemIndex]);
  if (POrigem.Opcao_tipo = numerica) and not (Key
in['0'..'9',',','.',Chr(8)]) then begin
     ShowMessage('Preencha o campo com um número');
     Key:= #0;
  end;
end;

{Não permite o preenchimento com um valor que não seja um número, ',' ou
'.' para uma variável númerica na caixa de valor2}
procedure TFrmMapa.EdValor2KeyPress(Sender: TObject; var Key: Char);
begin
     if not (Key in['0'..'9',',','.',Chr(8)]) then begin
        ShowMessage('Preencha o campo com um número');
        Key:= #0;
     end;
end;
{=========================================================================
 =========================================================================
                      COMPONENTE RICH_EDIT
 =========================================================================
 =========================================================================}
```

```
 {Procedimento que atualiza a Lista de Caminhos e o objeto RichEdit
 com o novo caminho encontrado}
procedure TFrmMapa.AtuJanRegras;
var TopoReg,PAuxReg,PAuxPilha:PTRPilhaRegra;
linha:integer;
begin
  TopoReg:=nil; {Utiliza uma outra pilha que terá o conteúdo invertido da
pilha original}
  PAuxPilha:=Topo; {Isso porque o caminho na pilha original vai do desti-
no para a origem}
  while(PAuxPilha<>nil) do begin
       new(PAuxReg);
       PAuxReg^.Nodo:=PAuxPilha^.Nodo;
       PAuxReg^.relacionamento1 :=PAuxPilha^.relacionamento1 ;
       PAuxReg^.nome_valor1:=PAuxPilha^.nome_valor1;
       PAuxReg^.relacionamento2 :=PAuxPilha^.relacionamento2 ;
       PAuxReg^.nome_valor2:=PAuxPilha^.nome_valor2;
       PAuxReg^.Prox:=TopoReg;
       TopoReg:=PAuxReg;
       PAuxPilha:=PAuxPilha^.Prox;
  end;
  PAuxReg:=TopoReg;
  {Esse laço executa a função que inclui o novo nodo na lista de regra e
   atualiza o RichEdit com o novo caminho}
   linha:=0;
   if PAuxReg.Prox= nil then exit;{Para não permitir regras vazias}
   FrmResultado.RERelatorio.Lines.Add('REGRA ' + IntToStr(numero));
   FrmResultado.RERelatorio.Lines.add('   SE');
  while(PAuxReg.Prox <>nil) do begin
             FrmResultado.RERelatorio.Lines.add('
'+TopoReg^.Nodo^.nome_nodo +
                    ' '+ TopoReg^.relacionamento1+ ' '+ Topo-
Reg^.nome_valor1 +
                    '     '+ TopoReg^.relacionamento2+ ' '+ Topo-
Reg^.nome_valor2);
        {Coloca premissas na regra}
       IncListaRegra(PrimListaRegra,UltListaRegra,linha,TopoReg^.Nodo);
       TopoReg:=PAuxReg^.Prox;
       Dispose(PAuxReg);
       PAuxReg:=TopoReg;
       if PAuxReg.Prox <>nil then
          FrmResultado.RERelatorio.Lines.Add('           E');
              {Atualiza a regra com nova premissa}
      if PAuxReg.Prox = nil then begin
          FrmResultado.RERelatorio.Lines.add('              ENTÃO');
          FrmResultado.RERelatorio.Lines.Add('              '+Resultado
+ ' = ' + TopoReg^.Nodo^.Nome_nodo);
          numero:=numero + 1;   {Incrementa o número de Ordem da Regra}
          {Se existir outra variável, coloca E}
        end;
   Linha :=Linha+1; {Atualiza o número de linhas da Pilha de regras}
   end;
   FrmResultado.RERelatorio.Lines.Add('');
 end;

{Procedimento para limpar a lista de regras}
procedure TFrmMapa.LimpaListaRegra(var PrimListaRegras:PTRListaRegra;
          var UltListaRegra:PTRListaRegra);
```

```
var PAux:PTRListaRegra;
begin
  PAux:=PrimListaRegras;
  while(PAux<>nil) do begin {Enquanto existir elementos na lista}
       PrimListaRegras:=PAux^.Prox;
       PAux^.Prox:=nil;
       Dispose(PAux);  {Desaloca}
       PAux:=PrimListaRegras;
  end;
  UltListaRegra:=PrimListaRegras; {null}

  DesMapa;
end;

{=========================================================================
 =========================================================================
                                    MENU
 =========================================================================
 =========================================================================}

 {Executado quando é selecionada a opção salvar no menu}
procedure TFrmMapa.Salvar1Click(Sender: TObject);
begin
  SalvarDialog.FileName:=NomeArq; {Nome do arquivo da caixa de diálogo
recebe o nome do arquivo atual}
  {Essa estrutura condicional existe porque se for selecionado <Cancelar>
na caixa de diálogo, o valor retornado será falso}
  if SalvarDialog.Execute then begin {Executa a caixa de diálogo Salvar-
Dialog}
     NomeArq:=SalvarDialog.FileName; {Nome do arquivo recebe nome do ar-
quivo selecionado no Dialogo}
     Salvar(NomeArq); {Procedimento de gravação no arquivo}
  end;
end;

{Procedimento que salva um arquivo tipado em disco.
 Descrição do formato do arquivo:
  - Primeiro todas as variáveis do grafo são gravadas em cada registro.
  - Após salvar todas as variáveis, começa a gravação dos caminhos, exe-
cutada do seguinte modo:
  - Depois do último registro, referente à última variai do grafo, é gra-
vado um registro com o seu campo CodNodo valendo o caractere '-'. Esse
caractere indica que o próximo registro é a variável origem e todos os
demais após o próximo e antes de outro '-' são adjacentes a essa variável
origem.
    É importante observar que nos registros, relativos os variáveis adja-
centes, são gravados apenas o código da variável, e não todos os dados
novamente (esses já estão nos primeiro registros). Esse formato economiza
    um considerável espaço em disco e tempo de gravação.
  - O arquivo termina quando não há nenhum registro depois do registro
com caractere '-'.
  - A extensão default do arquivo é <*.btc>, em homenagem à disciplina
Bases da Técnica Cirúrgica..  }
procedure TFrmMapa.Salvar(NomeArq:string);
var ArqMapa:Arquivo; {Cria uma variável para o arquivo}
    RegGrafo:Grafo;
    PAuxGrafo:PTRGrafo;
    PAuxAdj:PTRListaAdj;
begin
```

```
  AssignFile(ArqMapa,NomeArq); {Associa a variável de arquivo ao arquivo
selecionado}
  ReWrite(ArqMapa); {Ativa o arquivo para escrita}
  if IOResult<>0 then begin {Verifica se não ocorreu erro na abertura}
     ShowMessage('Erro na Abertura do Arquivo!');
     exit;
  end;
  PAuxGrafo:=PrimGrafo;
  while(PAuxGrafo<>nil) do begin {Grava todas as variáveis nos primeiros
registros}
        RegGrafo.CodNodo:=PAuxGrafo^.CodNodo;
        RegGrafo.nome_nodo:=PAuxGrafo^.nome_nodo;
        RegGrafo.opcao_tipo:=PAuxGrafo^.Opcao_tipo;
        RegGrafo.objetivo:=PAuxGrafo^.objetivo;
        RegGrafo.Pergunta:=PAuxGrafo^.Pergunta;
        RegGrafo.Motivo:=PAuxGrafo^.Motivo ;
        RegGrafo.relacionamento1:=PAuxGrafo^.relacionamento1 ;
        RegGrafo.Nome_valor1:=PAuxGrafo^.Nome_valor1 ;
        RegGrafo.relacionamento2:=PAuxGrafo^.relacionamento2 ;
        RegGrafo.Nome_valor2:=PAuxGrafo^.Nome_valor2 ;
        RegGrafo.xiNodo:=PAuxGrafo^.xiNodo;
        RegGrafo.yiNodo:=PAuxGrafo^.yiNodo;
        RegGrafo.xfNodo:=PAuxGrafo^.xfNodo;
        RegGrafo.yfNodo:=PAuxGrafo^.yfNodo;
        RegGrafo.Prox:=nil;
        RegGrafo.PrimAdj:=nil;
        RegGrafo.UltAdj:=nil;
        Write(ArqMapa,RegGrafo);
        PAuxGrafo:=PAuxGrafo^.Prox;
  end;
  PAuxGrafo:=PrimGrafo;
  while(PAuxGrafo<>nil) do begin {Começa novamente do início do grafo,
para gravar as adjacentes}
        RegGrafo.CodNodo:='-'; {Grava um registro com o identificador '-
'}
        Write(ArqMapa,RegGrafo);
        RegGrafo.CodNodo:=PAuxGrafo^.CodNodo; {Grava a variável origem}
        Write(ArqMapa,RegGrafo);
        PAuxAdj:=PAuxGrafo^.PrimAdj;
        while(PAuxAdj<>nil) do begin {Começa a gravação das adjacentes}
             RegGrafo.CodNodo:=PAuxAdj^.Nodo.CodNodo;
             RegGrafo.relacionamento1:=PAuxAdj^.relacionamento1 ;
             RegGrafo.nome_valor1:=PAuxAdj^.nome_valor1; {Grava o valor
na estrutura grafo}
             RegGrafo.relacionamento2:=PAuxAdj^.relacionamento2 ;
             RegGrafo.nome_valor2:=PAuxAdj^.nome_valor2; {Grava o valor
na estrutura grafo}
             Write(ArqMapa,RegGrafo);

             PAuxAdj:=PAuxAdj^.Prox;
        end;
        PAuxGrafo:=PAuxGrafo^.Prox;
  end;
  CloseFile(ArqMapa); {Fecha o arquivo}
end;

{Executado quando é selecionada a opção Abrir no menu}
procedure TFrmMapa.AbrirClick(Sender: TObject);
begin
```

```
  {Procedimento igual ao de salvar}
  AbrirDialog.FileName:=NomeArq;
  if AbrirDialog.Execute then begin
     NomeArq:=AbrirDialog.FileName;
     AbrirArq(NomeArq);
  end;
end;

{Procedimento para abertura do arquivo}
procedure TFrmMapa.AbrirArq(NomeArq:string);
var ArqMapa:Arquivo;
    RegGrafo:Grafo;
    PAuxGrafo,POrigem,PDestino:PTRGrafo;
begin
  AssignFile(ArqMapa,NomeArq); {Associa a variável de arquivo ao arquivo
selecionado}
  Reset(ArqMapa); {Ativa o arquivo para escrita}
  if IOResult<>0 then begin
     ShowMessage('Erro na Abertura do Arquivo!');
     exit;
  end;
  LimpaGrafo(PrimGrafo,UltGrafo); {Limpa o grafo que estiver na memória}
  LimpaListaOrigem(PrimListaOrigem,UltListaOrigem); {Limpa a Lista de
Origem}
  LimpaListaDestino(PrimListaDestino,UltListaDestino);{Limpa a Lista de
destino}
  LimpaListaRegra(PrimListaRegra,UltListaRegra); {Limpa a lista de regra}
  while not Eof(ArqMapa) do begin {Em quanto não for fim de arquivo}
        Read(ArqMapa,RegGrafo); {Lê o registro atual}
        if RegGrafo.CodNodo='-' then begin {Se o registro estiver identi-
ficado com '-', é sinal que começa as relações de adjacências}
           while not Eof(ArqMapa) do begin
                 if Eof(ArqMapa) then Break;
                 Read(ArqMapa,RegGrafo); {Lê a variável de origem}
                 POrigem:=AtribPtrGrafoCod(PrimGrafo,RegGrafo.CodNodo);
                 {Pesquisa o código lido e retorna um ponteiro para o
endereço onde as informações sobre esse nodo estão armazenadas}
                 if Eof(ArqMapa) then Break;
                 Read(ArqMapa,RegGrafo);
                 while(RegGrafo.CodNodo<>'-') do begin {Em quanto o re-
gistro não for '-', vai incluindo os nodos lidas na lista de adjacência
da variável origem}
                       PDesti-
no:=AtribPtrGrafoCod(PrimGrafo,RegGrafo.CodNodo);
                       {Pesquisa o código do nodo lido e retorna o seu
ponteiro}
                       AtribDistGra-
fo(POrigem,PDestino,RegGrafo.Relacionamento1,
                       RegGra-
fo.nome_valor1,RegGrafo.Relacionamento2,RegGrafo.nome_valor2);
                       {Executa o procedimento que liga dois nodos}
                       if Eof(ArqMapa) then Break;
                       Read(ArqMapa,RegGrafo);
                 end;
           end;
           end
        else begin {Se não for nodo adjacente}
           {Inclui um novo nodo no grafo}
           new(PAuxGrafo);
```

```
            PAuxGrafo^.CodNodo:=RegGrafo.CodNodo;
            PAuxGrafo^.nome_nodo:=RegGrafo.nome_nodo;
            PAuxGrafo^.opcao_tipo:=RegGrafo.opcao_tipo;
            PAuxGrafo^.objetivo:=RegGrafo.objetivo;
            PAuxGrafo^.Pergunta:=RegGrafo.Pergunta ;
            PAuxGrafo^.Motivo:=RegGrafo.Motivo ;
            PAuxGrafo^.relacionamento1:=RegGrafo.relacionamento1 ;
            PAuxGrafo^.Nome_Valor1:=RegGrafo.Nome_valor1 ;
            PAuxGrafo^.relacionamento2:=RegGrafo.relacionamento2 ;
            PAuxGrafo^.Nome_Valor2:=RegGrafo.Nome_valor2 ;
            PAuxGrafo^.xiNodo:=RegGrafo.xiNodo;
            PAuxGrafo^.yiNodo:=RegGrafo.yiNodo;
            PAuxGrafo^.xfNodo:=RegGrafo.xfNodo;
            PAuxGrafo^.yfNodo:=RegGrafo.yfNodo;
            PAuxGrafo^.Prox:=nil;
            PAuxGrafo^.PrimAdj:=nil;
            PAuxGrafo^.UltAdj:=nil;
            if PrimGrafo=nil then begin {Se for o primeiro}
               PrimGrafo:=PAuxGrafo;
               UltGrafo:=PAuxGrafo;
               end
            else begin
               UltGrafo^.Prox:=PAuxGrafo;
               UltGrafo:=PAuxGrafo;
            end;
         end;
  end;
  CloseFile(ArqMapa); {Fecha o arquivo}
  DesMapa; {Desenha o mapa}
  AtualizaCmbGrafo; {Atualiza os objetos}
end;


{Executado quando é selecionada a opção Novo no menu}
procedure TFrmMapa.NovoClick(Sender: TObject);
begin
  Saida;
end;

{Executado quando é selecionada a opção Imprimir no menu}
procedure TFrmMapa.ImprimirClick(Sender: TObject);
begin
  {Ativa a caixa de diálogo de impressão}
  if ImprimirDialog.Execute then FrmMapa.Print;
end;


{Executado quando é selecionada a opção Sair no menu}
procedure TFrmMapa.Sair1Click(Sender: TObject);
begin
  Saida;
  halt;
end;

end.
```