Universidade de São Paulo
Escola Politécnica da USP
Departamento de Engenharia Mecânica
Área de Automação e Sistemas

Trabalho de Graduação

RECONHECIMENTO DE PADRÕES EM TOMADA DE DECISÃO UTILIZANDO REDES NEURAIS ARTIFICIAIS

Autor : Emiliano G. de Castro
Orientador : Prof. Dr. Marcelo Godoy Simões
Data de Apresentação : 15-12-98

7,6 (sete e seis)

- São Paulo -
1998

Dedico este trabalho aos meus pais, que através de seus exemplos de vida sempre me mostraram o caminho certo a ser seguido.

AGRADECIMENTOS

Agradeço a todos os amigos, colegas de curso, professores e funcionários que contribuíram, direta ou indiretamente, para a realização deste trabalho.

Agradeço também a todas as pessoas que foram entrevistadas durante a pesquisa desenvolvida, pela infinita paciência. Sem elas este trabalho não teria sido possível.

Um agradecimento (e um pedido de desculpas) devem ser endereçados ao Alexandre Tebechrani, Michel Friedhofer, Marcelo Peçanha e Mauro Hiroshi Fujisawa, que pagaram caro o preço da minha amizade, preenchendo pesquisas de testes que pareciam nunca acabar.

Agradeço à Chris, pelo carinho sempre presente.

Aos professores Amilton Sinatora, Deniol Tanaka, Omar Moore de Madureira e Carlos Tu, do Departamento de Engenharia Mecânica da EPUSP, pela preciosa ajuda na elaboração das simulações.

Ao Prof. Aílton Amélio da Silva, do Instituto de Psicologia da USP, a quem eu credito o meu interesse pelo tema, pela paciente e constante colaboração no desenvolvimento da simulação mais difícil.

Agradeço especialmente ao Prof. Marcelo Godoy Simões, pela valiosa orientação em todas as áreas, sem a qual este trabalho jamais chegaria a lugar algum.

SUMÁRIO

1. INTRODUÇÃO

Sistemas especialistas são compreendidos como programas de computador interativos projetados para emular o processo de solução de problemas de um ou mais especialistas em uma particular área de conhecimento. O usuário típico de um sistema especialista não é um expert, e sim uma pessoa mais inexperiente. A idéia é que um pessoal menos experiente possa ter, na resolução de um problema, a performance de um grupo de especialistas.

Estes sistemas possuem uma base de conhecimento e um mecanismo de inferência, que serve para controlar o processamento desse conhecimento com base nos dados considerados. A saída é uma conclusão.

Sempre encarados como uma grande promessa, os sistemas especialistas tiveram, até agora, um limitado sucesso.

O decrescente custo de hardware e o crescente poder de software estão tornando cada dia mais viável a um profissional utilizar pacotes de técnicas de solução de problemas. Esses pacotes de software são hoje muito mais acessíveis do que há cinco anos atrás.

Existem exemplos claros de sucesso comercial. Por exemplo, um sistema de suporte à decisão utilizando procedimentos de otimização e sequenciamento ajudou o COI a agendar as datas e horários para os diferentes eventos das Olimpíadas de 1992. A Weyerhauser Corporation utiliza um sistema de suporte

à decisão empregando procedimentos de reconhecimento de padrões, otimização e simulação para cortar árvores em toras de vários tamanhos de forma a maximizar o lucro. O Citibank utiliza um sistema especialista para avaliar aplicações de empréstimo como um experiente administrador o faria. A Blue Cross utiliza um sistema especialista para processar reclamações de seguro. A Ford Motor Company utiliza um sistema especialista para diagnosticar problemas funcionais em seus robôs.

Entretanto, há indícios de que esses sistemas estão muito longe de atingir o seu potencial de mercado. Segundo Wolfgram:

"Muitos analistas industriais estimam que atualmente apenas 10% do potencial das aplicações de sistemas especialistas estão sendo reconhecidos..."

É nessa categoria que se encontram os tomadores de decisão. São sistemas que procuram emular, da forma mais fiel possível, o processo humano de decidir. Para tanto, consideram a idéia intuitiva de que o homem, ao considerar um conjunto de alternativas, de onde uma será escolhida para atingir de melhor maneira um objetivo especificado, tem a habilidade de avaliar, de maneira rápida e global, todas as alternativas frente a um conjunto de critérios (conscientes ou não). Uma vez feita essa complexa avaliação, surge a melhor alternativa, aquela que melhor se enquadrou dentro do que considerou como sendo a solução perfeita.

Esses sistemas, na tentativa de imitar a decisão humana, são obrigados a adotar algum tipo de modelo. Um exemplo, provavelmente o mais simples de todos, seria considerar uma somatória de notas ponderadas pela importância de cada critério.

No seguinte exemplo, são submetidos a escolha quatro concepções para um determinado produto industrial (um barbeador elétrico, por exemplo). Foram eleitos sete critérios para serem considerados no julgamento. Cada critério possui seu peso, associado à sua importância no processo de seleção. E cada alternativa possui uma nota relativa a cada critério. Esses dados compõem a chamada Matriz de Decisão a seguir :

CARACTERÍSTICAS	PESOS P	A Nota / N x P	B Nota / N x P	C Nota / N x P	D Nota / N x P
1) Desempenho	7	10 / 70	7 / 49	8 / 56	8 / 56
2) Aparência	8	7 / 56	9 / 72	10 / 80	7 / 56
3) Segurança	8	7 / 56	8 / 64	8 / 64	6 / 48
4) Durabilidade	6	10 / 60	9 / 54	7 / 42	10 / 60
5) Custo de Fabricação	9	7 / 63	8 / 72	9 / 81	10 / 90
6) Investimento Inicial	7	10 / 70	8 / 56	9 / 63	8 / 56
7) Prazo de Implantação	10	5 / 50	6 / 60	10 / 100	9 / 90
TOTAL N x P		**425**	**427**	**486**	**456**

Uma rápida análise desse exemplo já permite focar os principais problemas de um tomador de decisão.

Nesse exemplo foram abordados sete critérios de seleção. Alguns deles são essencialmente numéricos (custo de fabricação e investimento inicial, por exemplo) o que é bastante conveniente para a manipulação matemática. Porém, alguns critérios são qualitativos, de avaliação subjetiva. Nesses casos, cada

alternativa é avaliada frente a esses critérios e uma nota é a ela conferida. Obviamente, trata-se de uma fonte de incerteza no processo.

Outro aspecto bastante importante é revelado ao reparar nas pontuações finais. Embora a alternativa C seja a vencedora, existe entre ela e a alternativa D uma diferença de apenas 30 pontos. Não seria surpreendente se, na prática, o bom senso tivesse apontado a D como a melhor opção.

Uma reavaliação dos pesos poderia alterar completamente esse quadro. Nesses pesos estão representadas as sensibilidades de cada critério na decisão. E eles são informados diretamente pelo usuário, que estabelece suas importâncias relativas, de maneira completamente arbitrária. E aí reside um erro ainda mais perigoso.

Esse procedimento baseia-se no pressuposto de que as pessoas têm plena capacidade de análise de suas próprias decisões, capazes até de quantificar isso de maneira precisa. E ainda por cima, numa escala coerente com o modelo adotado pelo tomador de decisão. O que é uma premissa excessivamente otimista.

Nós temos domínio no sentido sintético de uma decisão. Nossas limitações surgem quando entram em cena um número muito grande de alternativas, critérios demais, ou então quando temos pouco tempo para a decisão. Porém, no sentido analítico, o máximo que podemos informar a respeito de uma decisão com o intuito de destrinchá-la será sempre um pouco mais que um palpite.

Essa dificuldade em ajustar um modelo matemático de maneira a conciliá-lo com o nosso bom senso é a principal responsável pelo fracasso de diversos sistemas. De acordo com Casey [1989, p.44] :

"Para cada história de sucesso, contabilizamos vários projetos de desenvolvimento de sistemas especialistas que falharam ou estão com sérios problemas. Muitos sistemas especialistas acabaram "mortos ao chegar" (nunca funcionaram), entre as filas de desempregados (nunca foram usados) ou cumprindo sentença em pesquisa e desenvolvimento (nunca foram terminados)."

Este trabalho propõe uma maneira de reduzir ao mínimo o conjunto de imprecisões que, de certa forma, condenam os sistemas de tomada de decisão (tomadores de decisão).

Ele parte do pressuposto de que um modelo matemático, desde que corretamente ajustado, pode emular uma decisão humana. Para tanto, não pode confiar cegamente na capacidade analítica humana.

Os modelos tradicionais, baseados na ponderação de critérios de diferentes importâncias relativas, que nos sistemas especialistas são fornecidos diretamente pelo usuário, são aqui substituídos por uma Rede Neural Artificial (RNA), que treinada com informações extraídas a partir da simulações (realizadas por meio de um conjunto de programas de pesquisa) de um processos de escolha, acaba por reconhecer e assimilar os padrões de decisão adotados pelos entrevistados.

Através de um mecanismo de aprendizado conhecido por retropropagação (*backpropagation*), esta rede ajusta seus ganhos intrínsecos de maneira iterativa. Esses ganhos são continuamente refinados até que a rede atinja uma configuração que melhor se enquadre dentro da lógica de escolha utilizada pela decisão humana.

Através de uma sequência de testes, é possível também avaliar a incerteza do processo, informação que permite ter uma idéia mais precisa da real viabilidade da utilização de um tomador de decisão.

Esse trabalho não tem, no entanto, a pretensão de desenvolver uma metodologia capaz de descobrir algo como uma "fórmula da decisão". Vale lembrar que qualquer modelo matemático, por mais elaborado que seja, jamais deixará de ser uma aproximação grosseira do altamente complexo mecanismo eletro-bioquímico que se realiza em nossos cérebros durante uma simples escolha.

O que ele faz é substituir a abordagem tradicional de forçar uma pessoa a analisar a sua lógica no processo de escolha por um sistema de reconhecimento de padrões de decisão baseado em uma tecnologia de Inteligência Artificial (RNA).

Devidamente treinado, este tomador de decisão tem a capacidade de emular, com alguma confiabilidade, um processo de escolha estritamente particular e específico, dentro de um limitado conjunto de alternativas e critérios. O que, considerando o panorama histórico descrito por Casey, pode ser considerado um significativo progresso.

2. REDES NEURAIS ARTIFICIAIS (RNA)

O mecanismo utilizado para reconhecimento de padrões em tomada de decisão é constituído por um software que emula uma Rede Neural Artificial.

Uma Rede Neural Artificial deve ser entendida como uma genuína imitação do que se acredita ocorrer na atividade cerebral humana, guardadas as devidas proporções. Uma rede neural processa informação mas não obedece a um algoritmo seqüencial. O processamento é baseado na decomposição paralela da informação complexa em elementos básicos.

Os seus elementos de processamentos são os neurônios artificiais, que inspirados nos seus correspondentes biológicos, possuem um conjunto de características familiares como entradas, forças sinápticas, potencial de ativação e saídas.

Os atributos básicos de uma Rede Neural Artificial podem ser divididos em duas categorias : arquitetura e neurodinâmica.

A arquitetura define a estrutura da rede, ou seja, o número de neurônios artificiais na rede, sua disposição em camadas e sua interconectividade.

A neurodinâmica consiste das propriedades funcionais da rede, o que representa o método através do qual a rede aprende, recorda, associa e continuamente compara nova informação com a base de conhecimento já adquirida, como ela classifica novas informações e como ela cria novas classificações se necessário.

A RNA implementada foi treinada com conjuntos de dados provenientes das simulações de processos de escolha.

2.1. Arquitetura da Rede

Quanto à arquitetura, a RNA pode ser descrita como uma rede de três camadas, com a seguinte composição:

- 5 neurônios na camada de entrada
- 3 neurônios na camada oculta
- 1 neurônio na camada de saída

O número de neurônios nas camadas de entrada e saída foi dimensionado de forma a se adequar às especificações dos conjuntos de treinamento, que têm uma única saída (a "nota" final de cada alternativa) como função de 5 variáveis de entrada (os critérios de decisão).

Foram realizados testes com conjuntos de treinamento para a escolha do número de neurônios na camada oculta. Partiu-se de uma regra empírica que recomenda que o número de neurônios da camada oculta esteja entre o número de neurônios da camada de entrada (5) e o número de neurônios da camada de saída (1). Dentre os valores 2,3 e 4 ensaiados, o melhor desempenho foi obtido pelas redes com 3 neurônios na camada oculta.

Quanto ao número de camadas ocultas (apenas uma), foi considerado que um superdimensionamento poderia comprometer a capacidade de generalização da rede, levando-a a "decorar" os resultados, ao invés de reconhecer um padrão dentro da lógica de decisão do entrevistado.

2.2. Neurodinâmica

Em relação à neurodinâmica da RNA, o método de aprendizado adotado foi o algoritmo de retropropagação (*backpropagation*), desenvolvido por Paul Werbos em 1974 e redescoberto independentemente por Rumelhart e Parker.

Desde o seu redescobrimento, o algoritmo de retropropagação tem sido amplamente utilizado como método de aprendizado em redes neurais multicamadas do tipo *feedforward*. O que diferencia esse algoritmo dos outros é que nele os pesos são calculados durante a fase de treinamento da rede.

Em geral, a dificuldade do treinamento em redes multicamadas consiste em calcular os pesos das camadas ocultas de uma maneira eficiente, tal que o erro na saída seja minimizado (ou zerado). Para atualizar esses erros, estes devem ser medidos. Na camada de saída isso é fácil, porém nas camadas internas não há observação direta do erro.

Durante o treinamento de uma rede, é apresentado um par de padrões $(\mathbf{X_k}, \mathbf{T_k})$, onde $\mathbf{X_k}$ é o padrão de entrada e $\mathbf{T_k}$ o objetivo ou padrão de saída desejado. O padrão $\mathbf{X_k}$ produz uma resposta na saída de cada neurônio em cada

camada, e assim, uma saída real $\mathbf{O_k}$ na camada de saída. Nessa camada, a diferença entre a saída real e a desejada produz um sinal de erro. Esse sinal de erro depende dos valores dos pesos dos neurônios em cada camada.

Esse erro é minimizado, e durante esse processo novos valores para os pesos são obtidos. A velocidade e precisão desse processo de aprendizado, isto é, do algoritmo de atualização dos pesos também depende de um fator conhecido como taxa de aprendizado.

O mecanismo de aprendizado pode ser mais rapidamente compreendido através da seguinte metáfora: dada uma certa estrutura de rede, podemos definir "configuração" como sendo cada combinação de valores possíveis para os pesos (ganhos) sinápticos de todos os neurônios da rede. Cada rede possui, então, infinitas configurações. A cada configuração corresponde um erro, que representa a incompatibilidade desta configuração com a função que desejamos ver aproximada (no caso, em reconhecer o padrão de decisão). Esse erro é então uma função de n variáveis (onde n é o número de pesos), que pode ser representada num espaço de dimensão $n+1$.

Para obter uma adequada visualização, podemos considerar uma rede hipotética contendo apenas dois pesos. Nesse caso, o erro é uma função de duas variáveis, e pode ser representado como uma superfície no espaço tridimensional, algo como a superfície terrestre numa região montanhosa, extremamente acidentada.

O mecanismo de aprendizado opera como uma bola que, jogada num ponto qualquer (aleatório) desta região montanhosa, é impelida pela força de

gravidade para a posição mais baixa o possível. O algoritmo calcula, a cada instante, a direção e o sentido que aponta para o gradiente da função erro (que indica o caminho para subir), e troca de configuração para outra no sentido oposto (que indica o caminho para descer). Esse processo é repetido até que a rede atinja uma configuração que minimiza a função erro.

Esse processo, porém, não é determinístico. Não é possível garantir que se atingiu a solução ótima, pois o algoritmo pode ter encontrado um mínimo local da função erro (e não o seu mínimo global). Como se a bola tivesse atingido o fundo de um vale, que não é o vale mais profundo da região. Como encontra subida para todos os lados, fica presa e não consegue atingir o verdadeiro ponto mais baixo da região (aliás nem sabe que ele existe).

Antes de iniciar um treinamento em retropropagação, devem ser considerados alguns parâmetros e estratégias, que têm como finalidade adequar o algoritmo aos conjuntos de treinamento e melhorar a velocidade de convergência da rede.

2.2.1. Função de Ativação

A função de ativação deve ser alguma forma de função não-linear. A adotada nesta implementação foi uma função sigmóide (logística), presente na função ***funcAtiv*** (vide Anexo – Programas Desenvolvidos – ERNA), expressa por :

2.2.2. Taxa de Aprendizado

Reflete o termo de velocidade do aprendizado. Se traduz na influência em que os pesos sinápticos são alterados em cada iteração. O valor adotado, após ensaios com alguns valores (entre 0 e 1), foi de 0,5 (variável ***taxa***).

2.2.3. Termo de Momento

Confere certa inércia à tendência de atualização dos pesos de cada neurônio, podendo ser interpretado como um coeficiente de suavizamento. Esse parâmetro se revelou extremamente eficiente para melhorar o desempenho da rede (redução do tempo de processamento através da redução do número de iterações até atingir o critério de parada). O valor que se mostrou mais eficiente foi de 0,2 (variável ***mom***).

2.2.4. Coeficiente de Arrefecimento

A estratégia utilizada consiste em verificar, depois de certo número de iterações, se a diferença entre o Erro Quadrático Médio (EQM) medido neste

instante e o EQM medido na última avaliação é menor que um milésimo do último EQM. O intervalo entre estas medições foi definido como meia-vida (identificado no software na variável ***meiaVida***), e foram adotados os valores de 50 e 100 iterações, dependendo do conjunto de treinamento.

Caso o EQM tenha variado menos que um milésimo neste intervalo, considerou-se que o treinamento não está evoluindo na velocidade esperada. Talvez o algoritmo já tenha atingido um valor de mínimo (local ou global). Mas existe a possibilidade da taxa de aprendizado e do termo de momento (inércia) estarem impedindo o algoritmo de realizar um ajuste mais "fino" nos pesos, com o objetivo de minimizar o erro global.

Isso pode ser resolvido multiplicando a taxa de aprendizado (***taxa***) e o termo de momento (***mom***) por um coeficiente de arrefecimento (***arr***). Assim o "passo" é reduzido, o que permite um refinamento maior na busca dos pesos sinápticos mais adequados. O valor adotado para esse coeficiente de arrefecimento foi de 0,98.

2.2.5. Critério de Parada

A estratégia abordada consiste em comparar, a cada meia-vida calculada, se o erro quadrático médio (***eqm***) é menor que a tolerância (***tol***) indicada no arquivo de configuração de treinamento (de 0,001 a 0,00001).

2.2.6. Pesos Sinápticos Iniciais

Nos primeiros ensaios do software, os pesos sinápticos assumiam, inicialmente, valores randômicos entre 0 e 1. Posteriormente, estes extremos foram reformulados para -0,1 e 0,1, com significativo efeito na performance da RNA.

2.2.7. *Thresholds*

Modelados como neurônios de que recebem sempre uma entrada constante (igual a 1), acrescentam um termo independente (offset) em todos os cálculos de somatórias ponderadas (sinapses). Tornam a rede mais "maleável" para se adaptar aos padrões de treinamento. Estão representados como um neurônio adicional na camada de entrada e outro na camada oculta (ambos com entrada constante e unitária).

2.3. Vantagens da utilização de RNA

2.3.1. Desconhecimento dos Fatores Relevantes

Numa abordagem estatística, seria indispensável que se conhecesse, a priori, quais fatores buscamos correlacionar. Uma grande vantagem das RNA é que elas são capazes de determinar quais dados são relevantes e quais não o são.

Os dados irrelevantes têm suas forças de conexão (pesos sinápticos) fortemente reduzidas, o que anula os seus efeitos.

2.3.2. Complexidade do Modelo

As RNA têm capacidade de lidar facilmente com um grande número de variáveis, algumas das quais produzindo efeitos pouco perceptíveis no resultado final. Mas o efeito agregado desses fatores de entrada consiste num modelo que se revela, frequentemente, mais acurado para problemas complexos do que qualquer modelo estatístico que possamos formular.

2.3.3. Alta Tolerância a Ruídos

Devido ao grande número de fatores de entrada, ruído nos dados não é um problema tão sério para RNA. A capacidade de aproximar um padrão acaba por considerar o comportamento médio da dinâmica dos dados, consequentemente anulando o efeito do ruído (que é essencialmente aleatório).

3. SIMULAÇÕES DE PROCESSOS DE ESCOLHA

Conforme apresentado no exemplo da introdução deste trabalho, alguns critérios são essencialmente numéricos (como custo de fabricação, temperatura, velocidade, tempo, etc.) e portanto são de fácil manipulação.

Porém outros (como aparência, segurança, sabor, conforto) não possuem uma escala ou instrumento de medição. Sua avaliação é tarefa trabalhosa, que requer o uso de ferramentas de Psicometria. Nesse trabalho, por simplificação de linguagem, os critérios do primeiro grupo serão chamados de quantitativos e os do segundo grupo de qualitativos.

Para verificar a abrangência do tomador de decisões foi testada a sua aplicabilidade em três processos de escolha de naturezas diferentes. Foram desenvolvidas três simulações, diferenciadas pela natureza de seus critérios. A simulação I contou apenas com critérios quantitativos; a II apenas com critérios qualitativos e a simulação III teve uma composição mista, contando com critérios quantitativos e qualitativos.

Cada simulação se constitui de um conjunto de três elementos, intimamente relacionados entre si : as alternativas, os critérios e o objetivo da escolha. A viabilidade do reconhecimento de padrões dependeu, em grande parte, da perfeita sintonia e coerência entre esses três elementos.

O objetivo da escolha não pode forçar o julgamento a levar em consideração algum critério que não tenha sido apresentado. Caso isso não seja

atendido, a importância desse critério não previsto aparecerá como um erro que dificultará o reconhecimento de um padrão.

É importante lembrar que os resultados fornecidos pela rede neural somente poderão ser corretamente interpretados se a decisão tiver alguma coerência racional. Assim sendo, devem ser eliminadas, na medida do possível, as fontes de julgamento inconsciente. Além do mais, uma RNA é eficaz para lidar com fenômenos essencialmente complexos, e não essencialmente aleatórios.

Os critérios eleitos devem, por sua vez, cobrir todos os aspectos das alternativas apresentadas no que se refere ao objetivo da decisão.

As alternativas também devem apresentar uma certa variabilidade em relação a todos os critérios. Seria impossível determinar a sensibilidade de um critério que não varia entre as alternativas.

Todas as alternativas apresentadas durante a simulação devem ser consideradas viáveis. Caso contrário, o entrevistado naturalmente descartará essa alternativa nas suas escolhas. Como isso tem um caráter eliminatório (ao invés do caráter classificatório, que é o que se busca aqui), seria outra fonte de erro.

Como ferramenta para atribuir uma escala às notas das alternativas em relação a cada critério considerado, foi utilizado um conjunto de programas especialmente desenvolvidos para uma pesquisa junto a um entrevistado.

Nestes programas, foi fundamental que a interface de operação tenha sido simples e intuitiva, o que foi conseguido utilizando interfaces fortemente visuais e operadas por mouse.

Nestes programas são apresentados aos entrevistados todas as combinações de valores (dois a dois) das alternativas frente ao critério em estudo. Em cada combinação, o entrevistado é requisitado a efetuar uma avaliação comparativa. Para escalonar as alternativas na avaliação global foi adotado o mesmo procedimento.

A seqüência em que essas combinações para avaliação comparativa são apresentadas obedece a técnicas psicométricas de compensação. Ao resultado dessas comparações é aplicado o método de Saaty para se obter uma escala numérica (normalizada).

Esta foi a metodologia que proporcionou os dados que, compilados, constituíram a base de informações para o reconhecimento de padrões em tomada de decisão. Uma parte destes dados foi aplicada no treinamento da RNA (conjunto de treinamento), e a outra nos testes para a medida de confiabilidade.

Por razões práticas, as simulações tiveram duas restrições : oito, dez ou doze alternativas e cinco critérios. Essa padronização facilitou inclusive uma comparação quanto a aplicabilidade do sistema em cada simulação. Em todos os casos, poderiam ter sido considerados um número maior de critérios necessários para a tomada de decisão, mas foram eleitos os cinco critérios mais críticos e os demais foram filtrados.

Dado o grau de importância dessa etapa, foram convidados a colaborar com a elaboração das simulações professores, de diferentes áreas, para cada simulação. Sem essa valiosa colaboração, esse trabalho seguramente teria um futuro duvidoso.

3.1. Simulação I : Seleção de Materiais para a Construção de um Vaso de Pressão

Nessa simulação foram apresentados oito alternativas de materiais (pré-definidos) que deveriam ser escolhidos para a construção de um vaso de pressão, especificamente um barril de chope. Os materiais descritos eram hipotéticos, para eliminar qualquer pré-julgamento ou conhecimento prévio de alguma característica não fornecida. Todos eles, no entanto, atendiam às especificações técnicas para a construção de um vaso de pressão, dispensando assim qualquer verificação quanto a critérios de falha, etc., de maneira que eles somente foram avaliados pelo bom senso do entrevistado.

A pesquisa foi realizada junto a especialistas da área de Engenharia de Materiais e alguns alunos de graduação em Engenharia Mecânica (área de Projeto e Fabricação).

A elaboração desta simulação contou com a colaboração do Prof. Amilton Sinatora e do Prof. Deniol Tanaka, do Departamento de Engenharia Mecânica da EPUSP.

Os critérios escolhidos (e seus respectivos conjuntos de valores para elaboração das funções de desejabilidade) foram:

- Densidade

- $0,9\ g/cm^3$

- 1,2 g/cm^3
- 2,7 g/cm^3
- 3,0 g/cm^3
- 8,0 g/cm^3

• Limite de Resistência

- 50 MPa
- 220 MPa
- 400 MPa
- 1500 MPa
- 2000 MPa

• Módulo de Elasticidade

- 1,5 GPa
- 70 GPa
- 80 GPa
- 200 GPa
- 220 GPa

• Tenacidade à Fratura

- 100 MPa • $\sqrt{m}$
- 30 MPa • $\sqrt{m}$
- 0,7 MPa • $\sqrt{m}$

- 5,0 MPa • √m
- 40 MPa • √m

- Custo

- US$ 0.2 / kg
- US$ 2.0 / kg
- US$ 8.0 / kg
- US$ 1.0 / kg
- US$ 50 / kg

3.2. Simulação II : Seleção de Parceiros para fins de Relacionamento Amoroso

Nessa simulação, as alternativas foram doze possíveis parceiros(as) (sorteados no início de cada pesquisa) para um relacionamento amoroso. O tema é objeto de pesquisa do Prof. Aílton Amélio da Silva, do Departamento de Psicologia Experimental do Instituto de Psicologia da USP, que colaborou ativamente na elaboração desta simulação.

Foram filtrados, na medida do possível, os critérios não eleitos na simulação, assim como também foi feito certo esforço no sentido de isolar cada critério nas pesquisas preliminares.

Para a pesquisa necessária à essa simulação foram observados todos os procedimentos éticos inerentes a uma pesquisa dessa natureza.

Nesta simulação foi considerado que mais importante que o valor absoluto de um determinado critério é o seu valor relativo ao próprio entrevistado. Neste caso, cada entrevistado é a sua própria "régua", devendo portanto fornecer alguns dados pessoais.

Foram entrevistados para esta simulação alunos do curso de graduação do Instituto de Psicologia da USP, sob a coordenação do Prof. Aílton Amélio da Silva.

Os critérios escolhidos (e seus respectivos conjuntos de valores para elaboração das funções de desejabilidade) foram:

- Altura

Cada entrevistado forneceu a sua própria altura, a altura ideal para um parceiro amoroso e seus limites máximos e mínimos desejáveis. Dentro destes limites, foram interpolados 5 valores.

- Idade

Assim como no caso da altura, também quanto à idade cada entrevistado forneceu a sua própria idade, a idade ideal para um parceiro amoroso e seus limites máximos e mínimos desejáveis. Dentro destes limites, foram interpolados 5 valores.

- Escolaridade

Cada entrevistado escolheu uma faixa de escolaridades desejáveis num parceiro amoroso que contivesse 4 valores consecutivos dentre a seguinte relação:

- Primeiro Grau Incompleto
- Primeiro Grau Completo
- Segundo Grau Incompleto
- Segundo Grau Completo
- Faculdade Incompleta
- Faculdade Completa
- Mestrado Incompleto

- Mestrado Completo
- Doutorado Incompleto
- Doutorado Completo

- Nível Sócio-econômico

Cada entrevistado escolheu uma faixa de níveis sócio-econômicos desejáveis num parceiro amoroso que contivesse 4 valores consecutivos dentre a seguinte relação:

- Classe Baixa
- Classe Média Baixa
- Classe Média Média
- Classe Média Alta
- Classe Alta

- Aparência Física

- Nada Bonita(o)
- Levemente Bonita(o)
- Medianamente Bonita(o)
- Muito Bonita(o)
- Extremamente Bonita(o)

3.3. Simulação III : Seleção de uma Concepção Técnica para um Veículo Utilitário de Entregas Urbanas

Essa simulação lidou com dez diferentes concepções técnicas (sorteadas no início de cada pesquisa), todas viáveis, de um veículo utilitário para entregas urbanas (ex : jornais, revistas, lavanderia). Neste caso, haviam quatro critérios quantitativos e um qualitativo.

A pesquisa foi realizada junto a alunos de graduação de Engenharia Mecânica.

Na elaboração dessa simulação colaborou o Prof. Omar Moore de Madureira e o Prof. Carlos Tu, do Departamento de Engenharia Mecânica da EPUSP.

Os critérios escolhidos (e seus respectivos conjuntos de valores para elaboração das funções de desejabilidade) foram:

- Aparência
  - Muito Feio
  - Feio
  - Médio
  - Bonito
  - Lindo

- Consumo

- 6,0 km/l
- 7,0 km/l
- 9,0 km/l
- 10,0 km/l
- 12,0 km/l

- Capacidade de Carga

- 1,0 ton.
- 1,2 ton.
- 1,5 ton.
- 1,8 ton.
- 2,0 ton.

- Preço

- R$ 14.000,00
- R$ 18.000,00
- R$ 23.000,00
- R$ 28.000,00
- R$ 32.000,00

- Manutenção

- R$ 1.000,00 / ano

- R$ 1.300,00 / ano
- R$ 1.500,00 / ano
- R$ 1.700,00 / ano
- R$ 2.000,00 / ano

4. PROGRAMAS DESENVOLVIDOS

O sistema de reconhecimento de padrões em tomada de decisão contou com um conjunto de programas desenvolvidos especialmente para este trabalho. Os programas principais foram:

- Pesquisa de Calibração do Método de Saaty (PCMS)
- Pesquisa de Definição das Funções de Desejabilidade (PDFD)
- Pesquisa de Escolha de Alternativas Viáveis (PEAV)
- Emulador de RNA (ERNA)

Além destes, foram desenvolvidos alguns outros programas acessórios, sem interface visual:

- Programa de Conversão de Dados de Pesquisa (PCDP)
- Programa Gerador de Configurações de Treinamento (PGCT)
- Programa Gerador de Treinamento de Grupo (PGTG)

apenas para conversão e formatação de dados.

Todos os programas foram desenvolvidos numa linguagem proprietária (Lingo®) de um software de autoria multimídia (Director® 6.0, da Macromedia®),

dentro de um paradigma de programação orientada à eventos. Os programas têm como plataforma operacional o Windows® 95, da Microsoft®.

A escolha deste sistema de desenvolvimento se deve, em grande parte, à importância de uma interface gráfica altamente intuitiva e amigável ao usuário, considerando que dos quatro principais programas desenvolvidos, três foram utilizados por entrevistados não familiarizados com o projeto (ou mesmo o ambiente Windows®).

Os três primeiros programas a serem descritos têm como função compor um conjunto (ou pelo menos parte de um conjunto) de treinamento para a RNA, que é desenvolvido num arquivo texto que parte de um formato básico, apresentado abaixo:

```
RECONHECIMENTO DE PADRÕES EM TOMADA DE DECISÃO UTILIZANDO REDES
NEURAIS ARTIFICIAIS

  Simulação 1 - Seleção de Materiais para a Construção de um Vaso de
Pressão

_____________________________________________

  Fase 1


Fator de Compensação z :
Erro Percentual Médio :

_____________________________________________

  Fase 2


    Escalas :

Densidade :

Limite de Resistência :

Módulo de Elasticidade :
```

Tenacidade à Fratura :

Custo :

Fase 3

Combinações : Todas

Notas :

Fim.

4.1. Pesquisa de Calibração do Método de Saaty (PCMS)

Este programa tem como finalidade familiarizar e treinar os entrevistados a utilizar o sistema de avaliação implementado.

Um mapa do estado norte-americano da Califórnia é apresentado e mostra assinaladas as posições de seis cidades: Eureka, San Francisco, Santa Cruz, Fresno, Santa Barbara e Los Angeles. Esta última cidade é assinalada em roxo, para destacá-la das demais, assinaladas em laranja.

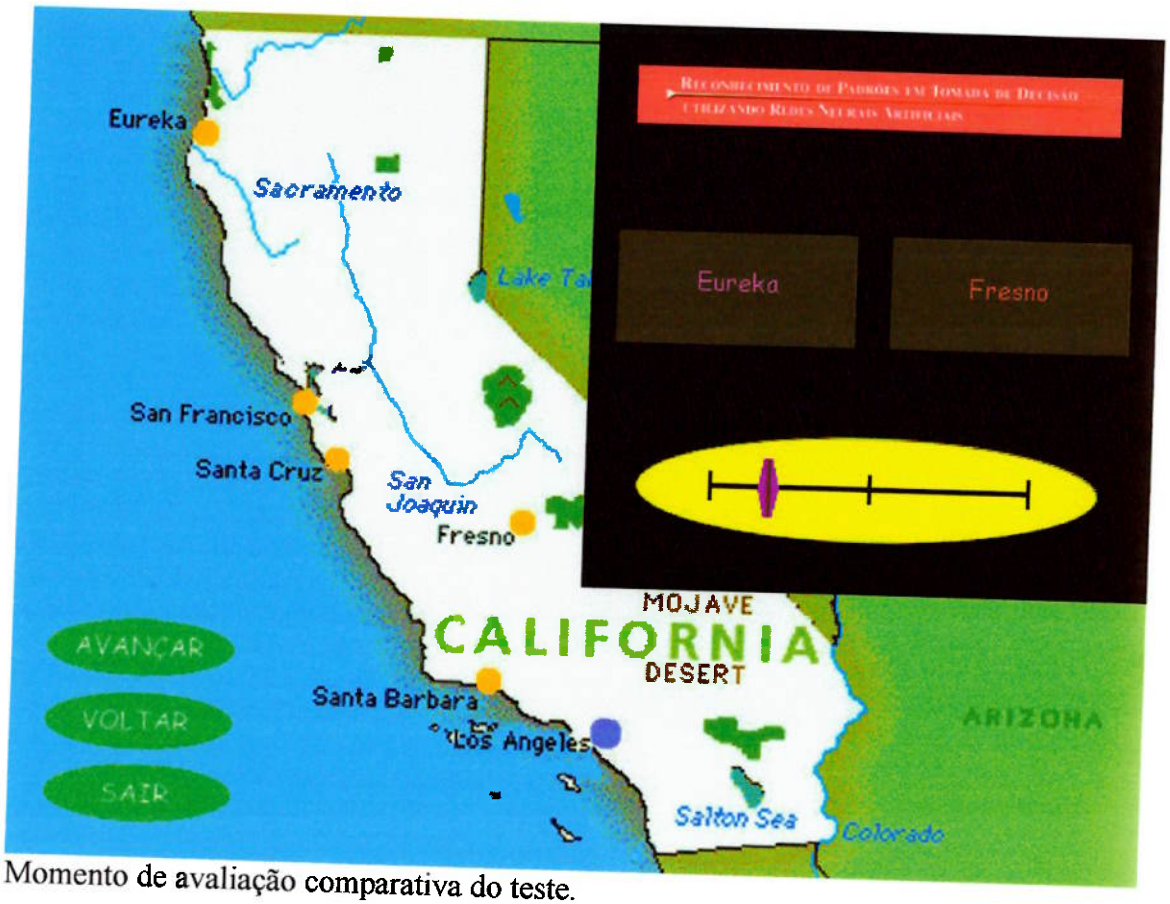
O objetivo do teste é verificar a capacidade de avaliação visual de distâncias do entrevistado. As cinco primeiras cidades são apresentadas emparelhadas, duas a duas, em todas as combinações possíveis (10), e o entrevistado deve assinalar, em cada comparação, qual das duas fica mais distante de Los Angeles.

Para tanto, deve posicionar um ponteiro ao longo de uma barra horizontal, que possui três marcações: o extremo esquerdo, o extremo direito e o centro.

A regra é a seguinte: o ponteiro deve ficar do lado da cidade mais distante. Quanto mais distante uma das cidades estiver de Los Angeles (em relação à outra, sempre), mais o ponteiro deve se aproximar do extremo lateral que representa essa cidade (extremo esquerdo para a cidade apresentada no lado esquerdo e vice-versa).

O ponteiro posicionado numa posição próxima ao centro indica que as cidades são quase equidistantes de Los Angeles. O empate exato (ponteiro no centro), no entanto, não é permitido. O entrevistado deve reavaliar cuidadosamente e se decidir pela distância que ele julgar maior.

Durante o teste, é apresentada a seguinte tela:



Momento de avaliação comparativa do teste.

Neste exemplo, estão sendo comparadas neste instante as cidades de Eureka (pela esquerda) e Fresno (pela direita). A barra está posicionada do lado esquerdo, o que indica que o entrevistado julgou que Eureka é mais distante de Los Angeles que Fresno. A posição polarizada (próxima ao extremo esquerdo)

do ponteiro mostra que este mesmo julgamento avaliou Eureka bem mais distante que Fresno.

O lado em que está o ponteiro indica qual cidade é mais distante, enquanto a distância do ponteiro ao centro da escala indica o quanto uma é mais distante que a outra. Este livre posicionamento dá margem a uma certa quantificação semântica (pouco mais distante, moderadamente mais distante, extremamente mais distante, etc.).

Resultados da Pesquisa de Calibração do Método de Saaty.

Após receber todas as avaliações comparativas possíveis, o software calcula a "escala" indicada pelo entrevistado, e a apresenta por meio de notas normalizadas.

O programa apresenta então uma comparação entre a escala estimada e a escala real (ambas normalizadas).

Além desta comparação, apresenta também dois importantes parâmetros fornecidos pela pesquisa:

- Fator de Compensação (z)

Indica o quanto o entrevistado foi econômico ou exagerado ao deslocar a barra. É natural que as pessoas não relacionem necessariamente as mesmas expressões semânticas (pouco mais distante, por exemplo) aos mesmos pontos na barra horizontal (distante 20% do centro, por exemplo). O que não traz absolutamente nenhum problema. Mas é importante que este fator seja conhecido para corrigir (padronizar) suas futuras avaliações, quando comparadas com as de outro entrevistado.

- Erro Quadrático Médio (EQM)

É uma medida de quão bem o entrevistado avaliou as distâncias relativas. Um erro muito grande indica que o entrevistado provavelmente não entendeu alguma regra do teste, e neste caso ele é instruído novamente, talvez até assistido

durante algumas comparações, e refaz o teste. Os erros mais frequentemente encontrados se situaram na faixa dos 2% a 8%, sendo tolerados até 12%.

É comum que se imagine que este erro indique quem é bom ou mal avaliador. O fato é que além da familiaridade com o método de avaliação adotado, entra no erro também a capacidade de avaliação visual de pequenas distâncias (já que as cidades estão representadas por pequenos pontos na tela do computador). E nada indica que exista alguma correlação entre avaliar visualmente bem pequenas distâncias e avaliar a desejabilidade de níveis sócio-econômicos para seleção de parceiros, por exemplo.

O processo de obtenção dos valores que compõem a escala é baseado no cálculo do autovetor da seguinte matriz:

Matriz de Avaliações Comparativas

$$\begin{pmatrix}
\frac{w_1}{w_1} & \frac{w_1}{w_2} & \frac{w_1}{w_3} & \frac{w_1}{w_4} & \frac{w_1}{w_5} \\
\frac{w_2}{w_1} & \frac{w_2}{w_2} & \frac{w_2}{w_3} & \frac{w_2}{w_4} & \frac{w_2}{w_5} \\
\frac{w_3}{w_1} & \frac{w_3}{w_2} & \frac{w_3}{w_3} & \frac{w_3}{w_4} & \frac{w_3}{w_5} \\
\frac{w_4}{w_1} & \frac{w_4}{w_2} & \frac{w_4}{w_3} & \frac{w_4}{w_4} & \frac{w_4}{w_5} \\
\frac{w_5}{w_1} & \frac{w_5}{w_2} & \frac{w_5}{w_3} & \frac{w_5}{w_4} & \frac{w_5}{w_5}
\end{pmatrix}$$

Cada elemento desta matriz representa o resultado de uma avaliação comparativa. Assim, $\mathbf{w_1/w_2}$ representa a "nota" da alternativa 1 dividida pela "nota" da alternativa 2.

Uma vez calculado este autovetor (vide Anexo – Programas Desenvolvidos – PCMS, rotina ***Saaty***), ele apresenta a sequência $[\mathbf{w_1,w_2,w_3,w_4,w_5}]$, que são os valores "absolutos" (normalizados) da escala, obtidos apenas a partir de vários valores relativos (das avaliações comparativas).

Este processo automatiza, de certa forma, o procedimento humano de compor uma escala subjetiva de valores. Ao nos depararmos com tal tarefa, primeiro observamos o conjunto como um todo, para termos uma noção de qual

elemento terá nota menor, qual terá nota maior, enfim, adquirirmos um termo de comparação.

A partir de então, realizamos sucessivas comparações para determinarmos uma espécie de "ranking" entre os elementos. Depois de pronto este "ranking", passamos a comparar novamente estes elementos com seus vizinhos no "ranking", de forma a obtermos uma noção de "distância" entre os valores (é quando verificamos, por exemplo, que a "distância" entre as notas do segundo e do terceiro colocados é muito menor do que entre as notas do terceiro e do quarto – o que significa que o segundo e o terceiro colocados estão quase empatados, e o quarto vem bem atrás).

Testes indicaram que os resultados obtidos através do Método de Saaty são geralmente muito mais acurados do que aqueles obtidos através deste processo natural.

O que ocorre no processo natural é que, na prática, não são realizadas todas as comparações possíveis, o que se mostraria extenuante e até pouco promissor, enquanto que pelo Método de Saaty as comparações são feitas entre todas as combinações possíveis. É possível que essa aparente redundância leve a uma avaliação mais criteriosa e bem feita. E é provavelmente aí que reside o "segredo do sucesso" deste método.

Os dois parâmetros calculados (fator de compensação e erro quadrático médio) são então armazenados em suas posições no arquivo texto que acompanhará todo o processo de pesquisa, como no exemplo a seguir.

```
RECONHECIMENTO DE PADRÕES EM TOMADA DE DECISÃO UTILIZANDO REDES
NEURAIS ARTIFICIAIS

  Simulação 1 - Seleção de Materiais para a Construção de um Vaso de
Pressão

  Fase 1

Fator de Compensação z : 0.59
Erro Percentual Médio : 13.51 %

  Fase 2

    Escalas :

Densidade :

Limite de Resistência :

Módulo de Elasticidade :

Tenacidade à Fratura :

Custo :

  Fase 3

Combinações : Todas

Notas :

Fim.
```

Rotinas científicas utilizadas:

montaMatriz : compõe uma matriz com valores de um vetor.

abreMatriz : inicializa uma matriz.

insere : insere um elemento numa matriz numa posição especificada.

lê : lê um elemento numa posição especificada de uma matriz.

escMat : escreve numa matriz específica.

leMat : lê numa matriz específica.

norm : normaliza o conjunto de valores de um vetor.

media : calcula a média do conjunto de valores de um vetor.

Saaty : devolve o autovetor (já normalizado) de uma matriz.

fit : calcula o fator de compensação z.

eleva : calcula a potenciação de um número por outro.

mmq : calcula o erro quadrático.

errPerc : calcula o erro percentual.

calc : gerencia o processo de cálculo.

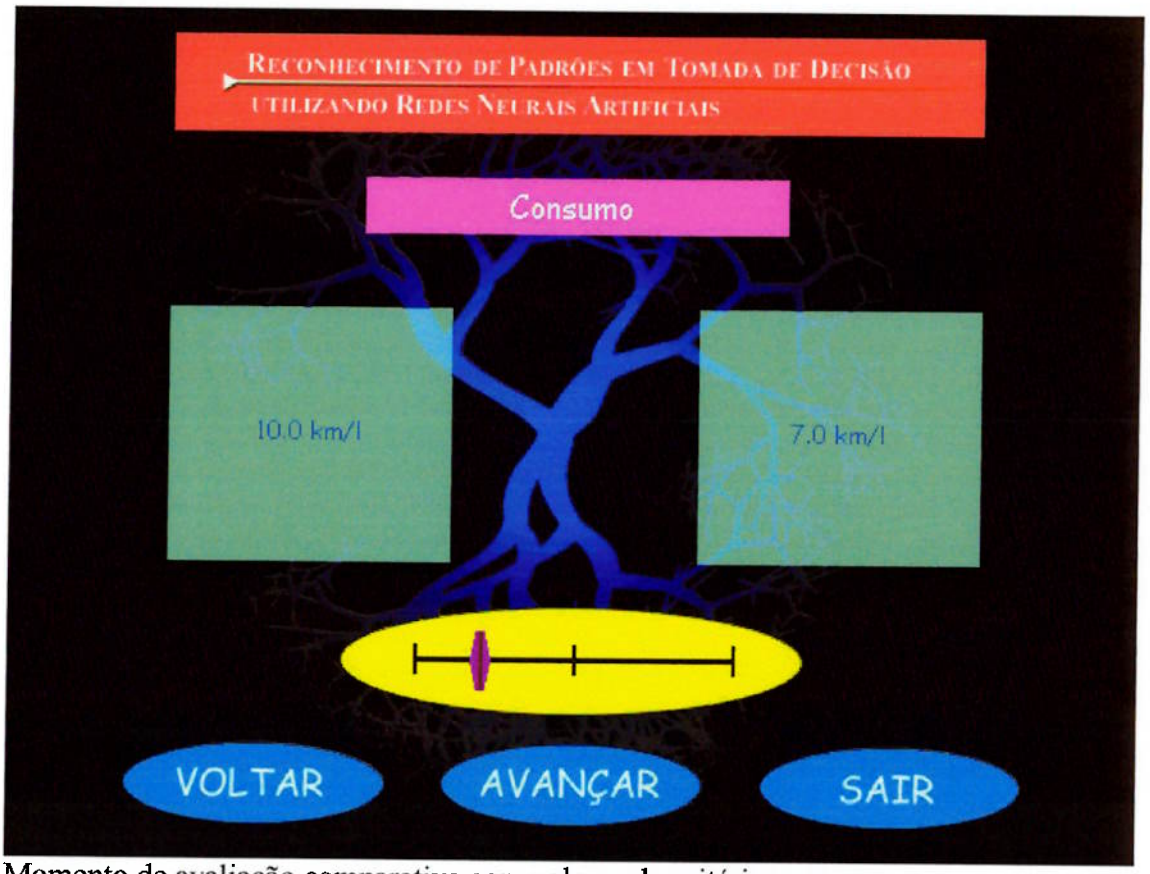
prepCalc : inicializa a rotina de gerenciamento.

4.2. Pesquisa de Definição das Funções de Desejabilidade (PDFD)

Neste programa é feita uma pesquisa de avaliação, semelhante à de Calibração do Método de Saaty, com a diferença de que agora o entrevistado, a essa altura já familiarizado com o método, avalia o conjunto de valores de cada critério, para obter o que foi denominado neste trabalho como "função de desejabilidade" de cada critério.

No exemplo abaixo, o entrevistado está avaliando uma das dez combinações possíveis entre os cinco valores de consumo de combustível. Após estas 10 avaliações, o programa calculará a "escala" deste entrevistado quanto ao critério consumo e procederá a pesquisa do critério subsequente (no caso capacidade de carga, preço, etc.).

Antes da avaliação de cada critério, são apresentados todos os valores possíveis que este critério irá assumir ao longo da pesquisa (para o estabelecimento do termo de comparação). Equivale à apresentação simultânea das cidades no mapa da Califórnia, na Pesquisa de Calibração do Método de Saaty.



Momento de avaliação comparativa com valores do critério consumo.

Após receber as avaliações comparativas de todos os critérios da simulação, o programa armazena os resultados do processamento dos dados coletados na pesquisa (as escalas dos critérios) nos seus respectivos lugares no arquivo texto que acompanha o processo de pesquisa.

```
RECONHECIMENTO DE PADRÕES EM TOMADA DE DECISÃO UTILIZANDO REDES
NEURAIS ARTIFICIAIS

  Simulação 1 - Seleção de Materiais para a Construção de um Vaso de
Pressão

---

  Fase 1

Fator de Compensação z : 0.59
```

```
Erro Percentual Médio : 13.51 %

-------------------------------------------------------------

  Fase 2

    Escalas :

Densidade : [0.057, 0.152, 0.232, 0.198, 0.360]

Limite de Resistência : [0.200, 0.407, 0.047, 0.245, 0.101]

Módulo de Elasticidade : [0.056, 0.146, 0.187, 0.568, 0.044]

Tenacidade à Fratura : [0.518, 0.147, 0.032, 0.097, 0.206]

Custo : [0.506, 0.149, 0.085, 0.231, 0.028]

-------------------------------------------------------------

  Fase 3

Combinações : Todas

Notas :

-------------------------------------------------------------

Fim.

-------------------------------------------------------------
```

Rotinas científicas utilizadas:

inicializa : prepara um vetor que armazenará as posições dos ponteiros movidos pelo entrevistado.

montaMatriz : compõe uma matriz com valores de um vetor.

abreMatriz : inicializa uma matriz.

insere : insere um elemento numa matriz numa posição especificada.

lê : lê um elemento numa posição especificada de uma matriz.

escMat : escreve numa matriz específica.

leMat : lê numa matriz específica.

norm : normaliza o conjunto de valores de um vetor.

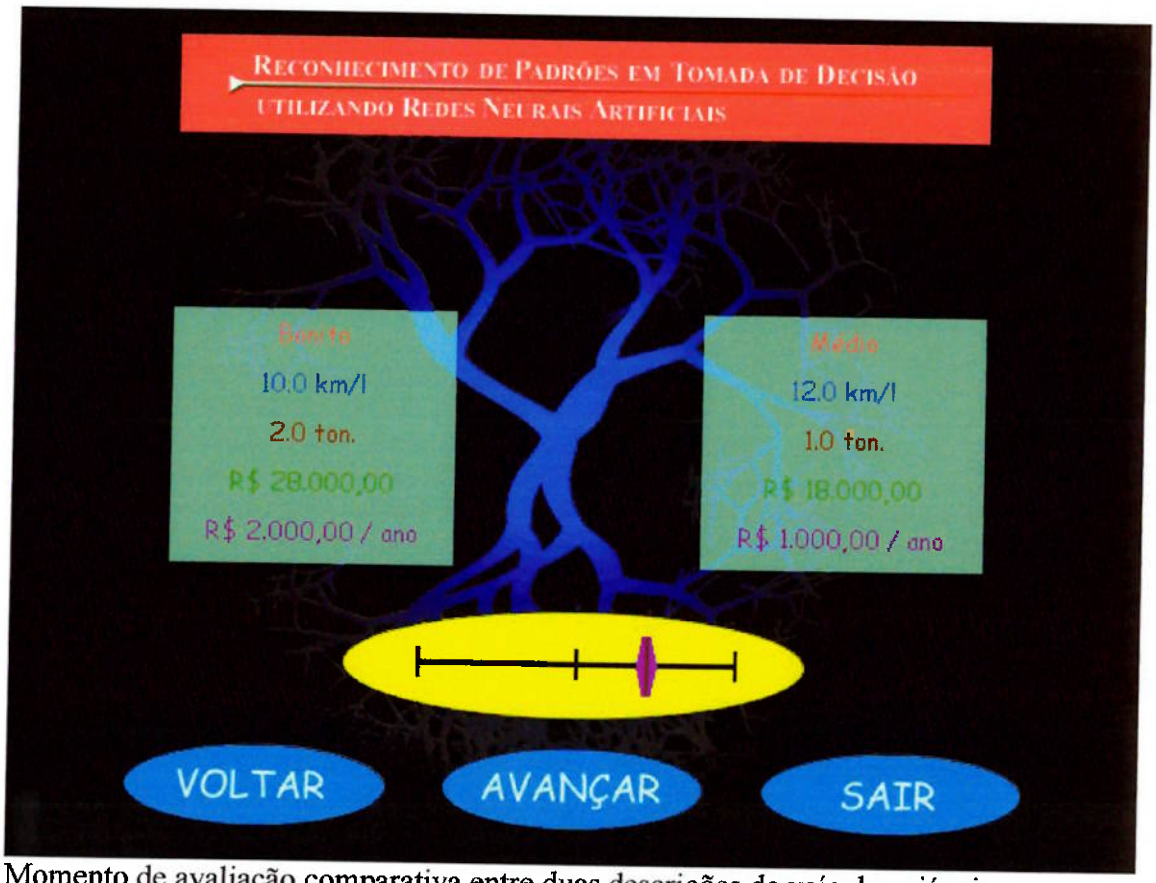
media : calcula a média do conjunto de valores de um vetor.

Saaty : devolve o autovetor (já normalizado) de uma matriz.

4.3. Pesquisa de Escolha de Alternativas Viáveis (PEAV)

Neste programa, o entrevistado finalmente avalia entre alternativas (candidatos) completas. São descritas alternativas viáveis para o objetivo da escolha através da apresentação de seu conjunto de características: um valor para cada critério de avaliação.

No exemplo a seguir, o entrevistado está comparando dois veículos hipotéticos quanto aos cinco critérios de avaliação.



Momento de avaliação comparativa entre duas descrições de veículos viáveis.

O arquivo texto que acompanha a pesquisa traz a indicação de quais combinações serão comparadas nesta fase. Caso haja a indicação de "todas" (que é o caso do arquivo texto em questão), são utilizadas oito combinações pré-definidas para compor o conjunto de escolha. Após a pesquisa, o resultado (o conjunto de notas dos "candidatos") é armazenado no item "Notas".

```
RECONHECIMENTO DE PADRÕES EM TOMADA DE DECISÃO UTILIZANDO REDES
NEURAIS ARTIFICIAIS

  Simulação 1 - Seleção de Materiais para a Construção de um Vaso de
Pressão

---

  Fase 1
```

```
Fator de Compensação z : 0.59
Erro Percentual Médio : 13.51 %

---------------------------------------------------------

  Fase 2

    Escalas :

Densidade : [0.057, 0.152, 0.232, 0.198, 0.360]

Limite de Resistência : [0.200, 0.407, 0.047, 0.245, 0.101]

Módulo de Elasticidade : [0.056, 0.146, 0.187, 0.568, 0.044]

Tenacidade à Fratura : [0.518, 0.147, 0.032, 0.097, 0.206]

Custo : [0.506, 0.149, 0.085, 0.231, 0.028]

---------------------------------------------------------

  Fase 3

Combinações : Todas

Notas : [0.154, 0.126, 0.137, 0.072, 0.219, 0.137, 0.090, 0.066]

---------------------------------------------------------

Fim.
```

Já no arquivo texto abaixo, referente a outra simulação, são indicadas quais combinações utilizar nas comparações dessa fase.

```
RECONHECIMENTO DE PADRÕES EM TOMADA DE DECISÃO UTILIZANDO REDES
NEURAIS ARTIFICIAIS

  Simulação 3 - Seleção de Veículo Utilitário para Entregas Urbanas

---------------------------------------------------------

  Fase 1
```

```
Fator de Compensação z : 0.62
Erro Percentual Médio : 5.95 %

_____________________________________________

  Fase 2

    Escalas :

Aparência : [0.037, 0.080, 0.147, 0.238, 0.499]

Consumo : [0.038, 0.060, 0.150, 0.190, 0.561]

CapacidadeDeCarga : [0.040, 0.069, 0.145, 0.269, 0.477]

Preço : [0.532, 0.252, 0.131, 0.056, 0.029]

Manutenção : [0.557, 0.226, 0.116, 0.065, 0.036]

_____________________________________________

  Fase 3

Combinações : [[2, 2, 4, 2, 5], [3, 5, 2, 1, 4], [5, 1, 3, 4, 2], [4,
3, 1, 5, 1], [1, 4, 5, 3, 3], [3, 2, 4, 2, 1], [2, 5, 3, 1, 3], [4, 4,
2, 5, 2], [1, 1, 1, 4, 5], [5, 3, 5, 3, 4]]

Notas : [0.056, 0.144, 0.029, 0.040, 0.157, 0.136, 0.274, 0.051,
0.015, 0.097]

_____________________________________________

Fim.

_____________________________________________
```

Em algumas pesquisas, esta etapa foi repetida várias vezes, com o objetivo de compor um conjunto de treinamento com maior número de exemplos, assim como obter dados adicionais para teste.

Nestes casos, para que os dados pudessem ser agrupados sem distorção de escala, o programa forçava a inclusão de duas alternativas particulares em todos os conjuntos de combinações: a pior e a melhor possível, considerando as funções de desejabilidade do entrevistado. Estas alternativas se comportavam

como medidas de referência, como termos de comparação para que alternativas de grupos de combinações diferentes pudessem ser reunidas num só grupo.

Rotinas científicas utilizadas:

inicializa : prepara um vetor que armazenará as posições dos ponteiros movidos pelo entrevistado.

montaMatriz : compõe uma matriz com valores de um vetor.

abreMatriz : inicializa uma matriz.

insere : insere um elemento numa matriz numa posição especificada.

lê : lê um elemento numa posição especificada de uma matriz.

escMat : escreve numa matriz específica.

leMat : lê numa matriz específica.

norm : normaliza o conjunto de valores de um vetor.

media : calcula a média do conjunto de valores de um vetor.

Saaty : devolve o autovetor (já normalizado) de uma matriz.

sorteiaCand : sorteia composições de candidatos viáveis para o processo de seleção.

4.4. Programa de Conversão de Dados de Pesquisa (PCDP)

A simulação de Seleção de Parceiros contou com a valiosa colaboração de uma turma de alunos de graduação do Instituto de Psicologia da USP. Devido ao grande número de entrevistados, não foi possível aplicar a pesquisa via software.

Foram distribuídos formulários impressos, contendo exatamente as mesmas informações que as apresentadas na tela do computador. Os entrevistados tiveram que assinalar a lápis a posição do ponteiro em cada barra horizontal impressa no formulário.

Isso trouxe uma trabalho e uma dificuldade adicional. Trabalho para tabular todos os dados de cerca de 300 páginas de formulários, medindo com uma régua, uma a uma, a posição de cerca de 3.000 marcas em barras horizontais.

Estes medidas (em cm) tiveram então que ser digitadas em arquivos textos e convertidas para se adequar à escala utilizada no resto dos programas (valores entre –1 e +1).

Essa conversão foi efetuada por este programa, que desprovido de interface de operação, apenas lê dados de um arquivo texto, processa-os e armazena os resultados em outro arquivo texto.

Rotinas científicas utilizadas:

montaMatriz : compõe uma matriz com valores de um vetor.

abreMatriz : inicializa uma matriz.

insere : insere um elemento numa matriz numa posição especificada.

lê : lê um elemento numa posição especificada de uma matriz.

converte : gerencia o processo de conversão.

automatiza : coordena um conjunto de conversões.

escMat : escreve numa matriz específica.

leMat : lê numa matriz específica.

norm : normaliza o conjunto de valores de um vetor.

media : calcula a média do conjunto de valores de um vetor.

4.5. Programa Gerador de Configurações de Treinamento (PGCT)

Os arquivos texto que acompanharam as pesquisas possuem, ao final, todas as informações necessárias para o treinamento da RNA. No entanto, estas informações estão dispostas num formato pouco prático.

Este programa, desprovido de interface de operação, apenas lê dados de um arquivo texto e formata-os para um arquivo de treinamento.

Os dados armazenados no arquivo texto apresentado até aqui foram formatados e armazenados em outro arquivo texto, apresentado a seguir:

```
RECONHECIMENTO DE PADRÕES EM TOMADA DE DECISÃO UTILIZANDO REDES
NEURAIS ARTIFICIAIS

Arquivo de Treinamento

Titulo : SeleMat - Prof. Sinatora

Taxa de Aprendizado : 0.5
Momento : 0.2
Tolerancia : 0.0001
Numero de Entradas : 5
Numero de Saidas : 1
Numero de Exemplos : 5

i1 = [0.057,0.152,0.232,0.198,0.360]

i2 = [0.200,0.407,0.047,0.245,0.101]

i3 = [0.056,0.146,0.187,0.568,0.044]

i4 = [0.518,0.147,0.032,0.097,0.206]

i5 = [0.506,0.149,0.085,0.231,0.028]
```

```
o = [0.154,0.137,0.219,0.137,0.066]
Fim
```

Rotina científica utilizada:

compila : formata os dados proveniente do arquivo texto que acompanhou a pesquisa num arquivo de treinamento para o Emulador da RNA.

4.6. Programa Gerador de Treinamento de Grupo (PGTG)

Foram agrupados diversos conjuntos de treinamento, de diferentes entrevistados, para verificar se havia algum tipo de "padrão de escolha" comum, e também para analisar se havia algum tipo de comportamento médio.

Este programa, desprovido de interface de operação, apenas lê dados de um grupo de arquivos texto e formata-os para um único arquivo de treinamento.

Rotina científica utilizada:

compila : formata os dados proveniente dos arquivos texto que acompanharam a pesquisa de um grupo de entrevistados num único arquivo de treinamento para o Emulador da RNA.

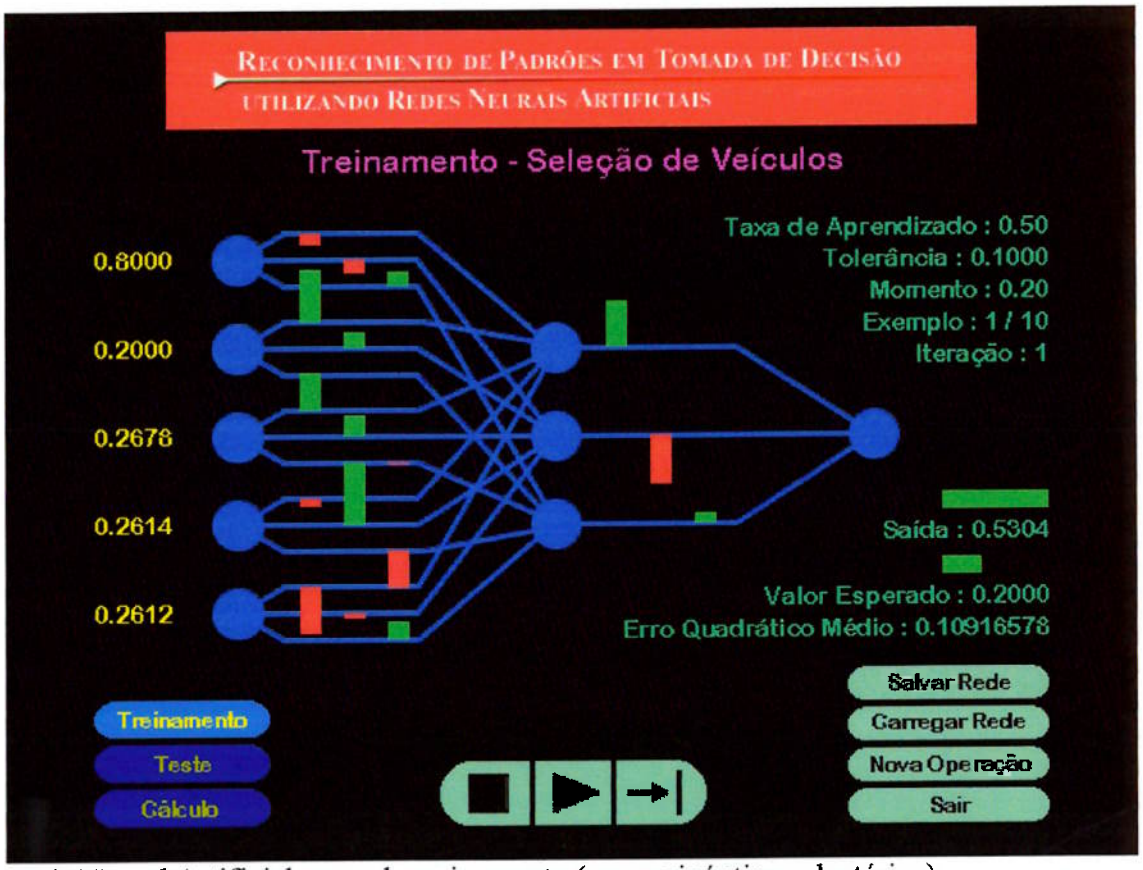
4.7. Emulador de RNA (ERNA)

O programa Emulador de Rede Neural Artificial é composto por uma interface de operação extremamente visual, projetada de forma a possibilitar o acompanhamento de todas as variáveis envolvidas na dinâmica de operações da RNA.

A arquitetura da RNA fica exposta num diagrama esquemático, com círculos representando os neurônios e linhas representando as suas ligações, as sinapses. As barras coloridas posicionadas sobre estas ligações representam os pesos (ganhos) sinápticos. Foi adotada uma convenção de verde para valores positivos e vermelho para negativos.

As entradas são apresentadas ao lado dos neurônios da camada de entrada, enquanto a saída calculada e esperada ficam ao lado do neurônio da camada de saída.

Os parâmetros de neurodinâmica que são variáveis (taxa de aprendizado e momento, devido ao coeficiente de arrefecimento), podem ser monitorados constantemente, assim como o valor de tolerância (que representa o critério de parada) na parte direita da tela.



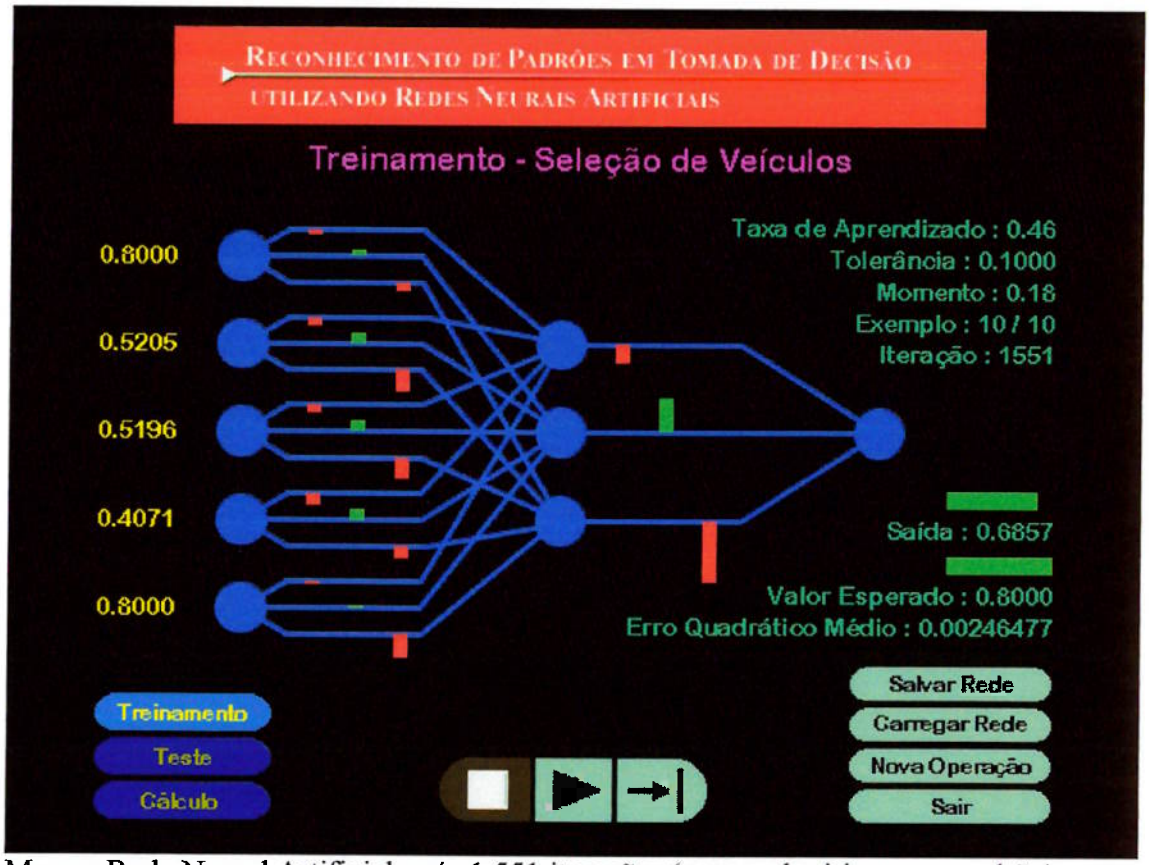
Rede Neural Artificial antes do treinamento (pesos sinápticos aleatórios).

Também pode ser verificada a cada instante qual a posição no processo de treinamento. É possível acompanhar qual é o número do exemplo sendo apresentado à RNA, assim como qual é o número da iteração sendo feita no momento.

O programa possibilita três modos de operação:

- **Treinamento** : sua principal função. Processa o algoritmo de aprendizado por retropropagação *(backpropagation)*. Os pesos (ganhos) sinápticos são constantemente modificados, o que pode ser acompanhado graficamente no diagrama apresentado na tela.

- **Teste** : verifica a eficácia do padrão assimilado pela RNA após o seu treinamento, comparando o resultado que a rede fornece a um conjunto de entradas de saídas conhecidas.
- **Cálculo** : calcula as respostas que a rede fornece a um conjunto de entradas (de saída desconhecida), acreditando que a rede já foi treinada o bastante para assimilar o padrão de escolha desejado.



Mesma Rede Neural Artificial após 1.551 iterações (pesos adquiriram um padrão).

O programa oferece um intuitivo sistema de controle:

- *Stop* : interrompe o processo que está sendo realizado.
- *Play* : procede a operação, um exemplo de cada vez, para possibilitar um atento acompanhamento da dinâmica do algoritmo.
- *Fast-forward* : dispara a operação até o seu fim (até satisfazer o critério de parada, no caso de treinamento, ou acabar de calcular o conjunto de dados, nos casos de teste e cálculo).

O programa permite salvar uma configuração atingida pela RNA para ser carregada posteriormente.

Existe a possibilidade de acompanhar o processamento dentro da RNA, habilitando (CTRL-L) um sistema de "iluminação" que acompanha o percurso do processamento, indo e vindo ao longo do algoritmo de treinamento. Altamente didático, tem a desvantagem de comprometer seriamente a velocidade de processamento.

Caso o usuário esteja realmente preocupado com velocidade de processamento, ele pode desabilitar (CTRL-E) a apresentação *real-time* dos pesos (ganhos) sinápticos e demais variáveis, atualizando-as na tela somente a cada meia-vida (período definido no arquivo de configuração de treinamento, geralmente de 50 ou 100 iterações). Há um enorme incremento de velocidade.

Tendo em vista que o tempo de convergência para um grande conjunto de treinamento é em geral grande (os maiores conjuntos apresentaram tempos de convergência de 20 minutos a 4 horas), foi implementado um sistema de alarme sonoro que, habilitado pelo usuário (CTRL-S), avisa quando a rede convergiu.

Rotinas científicas utilizadas:

sorteiaSeq : sorteia a sequência em que os elementos do conjunto de treinamento serão apresentados à RNA, em cada iteração (a mesma sequência pode resultar num padrão repetitivo de atualização de pesos sinápticos, o que reduziria a performance do algoritmo de treinamento).

sorteiaW : sorteia o valor inicial dos pesos, entre –0,1 e 0,1.

preparaIteracao : inicializa a rotina de "loop" de iteração.

preparaBP : inicializa a rotina que gerencia o algoritmo de treinamento por retropropagação *(backpropagation)*.

BP : gerencia o algoritmo de treinamento por retropropagação *(backpropagation)*.

funcAtiv : rotina que retorna a função sigmóide do valor passado à ela.

normaliza : encontra os valores de máximo e mínimo de uma entrada. Reescala então todos os valores desta entrada de forma que o seu novo mínimo seja 0,2 e seu novo máximo seja 0,8. Realiza essa operação com todas as entradas e também com a saída. Acarreta um brutal ganho de performance no treinamento.

preparaTeste : inicializa a rotina que gerencia o algoritmo de teste.

Teste : gerencia o algoritmo de teste (análogo ao da rotina ***BP***, com a diferença que só se processa durante uma única iteração e apenas no sentido entrada→saída). Ao invés de treinar a rede, compara as respostas que a rede

fornece a um conjunto de entradas com as saídas conhecidas deste mesmo conjunto.

normalizaTeste : encontra os valores de máximo e mínimo de uma entrada. Reescala então todos os valores desta entrada de forma que o seu novo mínimo seja 0,2 e seu novo máximo seja 0,8. Realiza essa operação com todas as entradas e também com a saída.

preparaCalculo : inicializa a rotina que gerencia o algoritmo de cálculo.

calculo : gerencia o algoritmo de cálculo (análogo ao da rotina de teste, com a diferença que desconhece os valores de saída). Ao invés de treinar ou testar a rede, calcula as respostas que a rede fornece a um conjunto de entradas, acreditando que a rede já foi treinada o bastante para assimilar o padrão de escolha desejado.

normalizaCalculo : encontra os valores de máximo e mínimo de uma entrada. Reescala então todos os valores desta entrada de forma que o seu novo mínimo seja 0,2 e seu novo máximo seja 0,8. Realiza essa operação com todas as entradas (não há saída conhecida no algoritmo de cálculo).

5. RESULTADOS

Os programas de pesquisa foram aplicados no público-alvo pré-estabelecido que incluiu, para cada simulação:

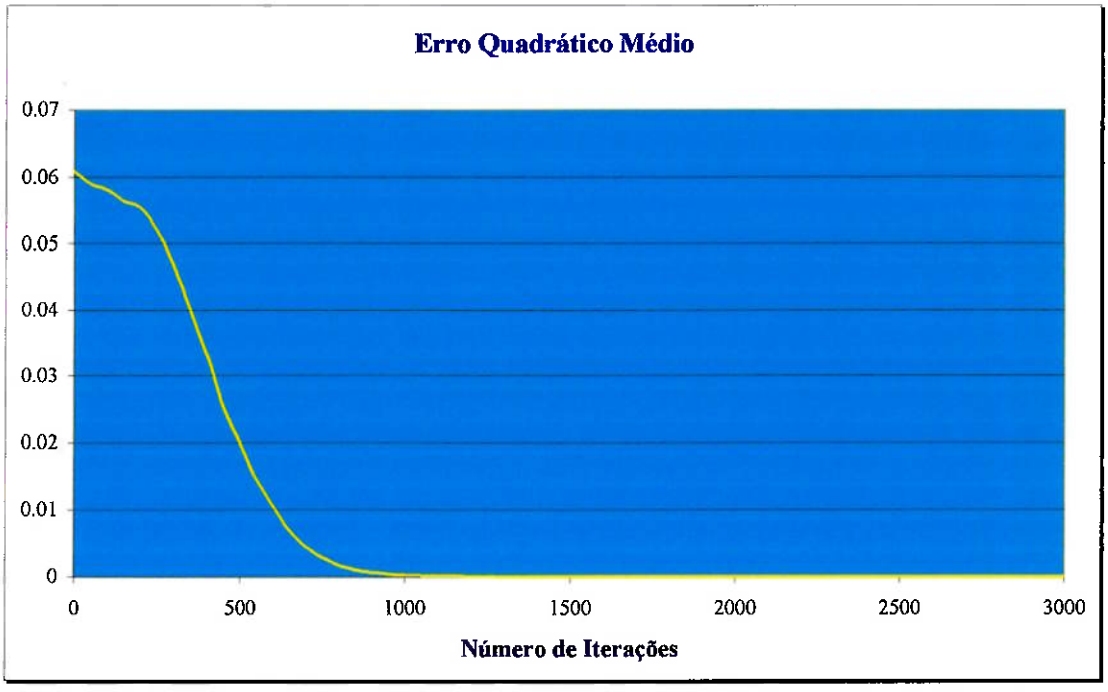
- Seleção de Materiais : 12 entrevistados
- Seleção de Parceiros : 46 entrevistados (17 homens e 29 mulheres)
- Seleção de Veículos : 21 entrevistados

Cada entrevistado forneceu pelo menos um conjunto de treinamento completo. Foi treinada uma rede para cada entrevistado, num total de 79 redes. Destas, apenas três (duas de seleção de materiais e uma de seleção de parceiros) não convergiram, o que resulta num aproveitamento de 96% quanto à capacidade do sistema em reconhecer padrões.

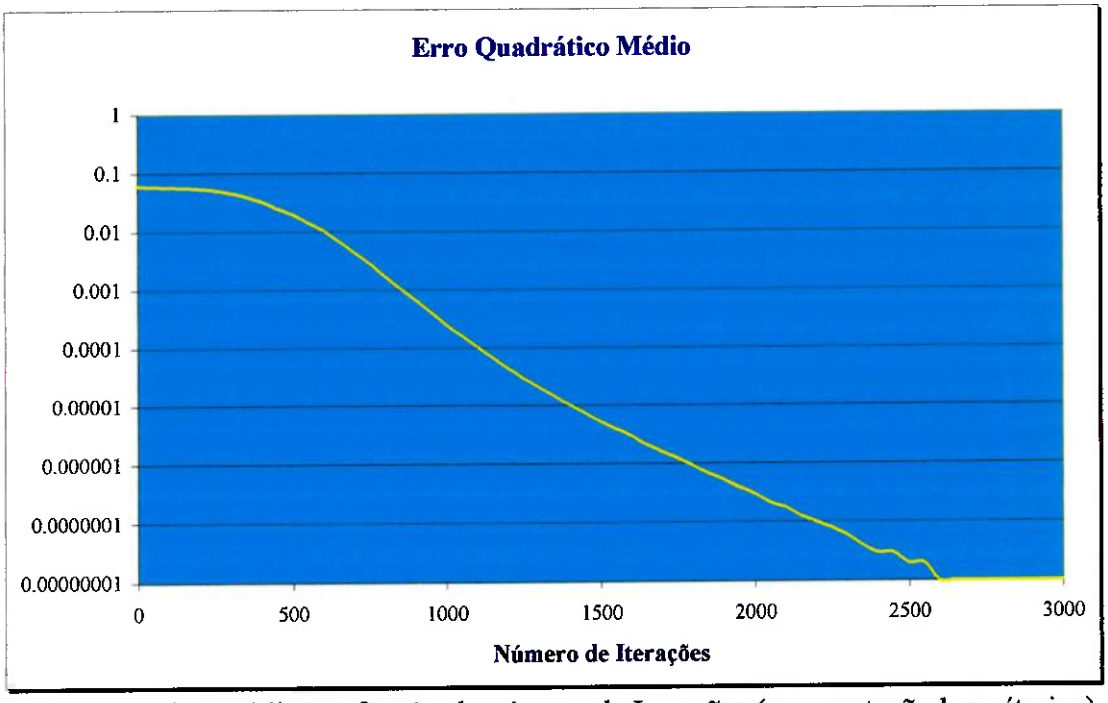
Além destes conjuntos de treinamento individuais, foram formatados conjuntos de treinamento reunindo todos os entrevistados de cada grupo, resultando em mais quatro redes (uma de seleção de materiais, uma de seleção de veículos e duas de seleção de parceiros – uma para os homens e outra para as mulheres). Destas quatro redes, todas convergiram.

Todas estas redes foram treinadas para atingir uma tolerância de 0,001 ($1{,}0 \times 10^{-3}$).

O número de iterações necessárias para a convergência variou bastante de treino para treino, porém em todas as redes que foram analisadas mais detalhadamente foi possível observar que o comportamento do Erro Quadrático Médio ao longo das iterações foi de decaimento praticamente regular, conforme pode ser verificado nos gráficos a seguir, relativos a um conjunto de treinamento proveniente de uma das pesquisas da simulação de seleção de materiais.



Erro Quadrático Médio em função do número de Iterações (apresentação linear).

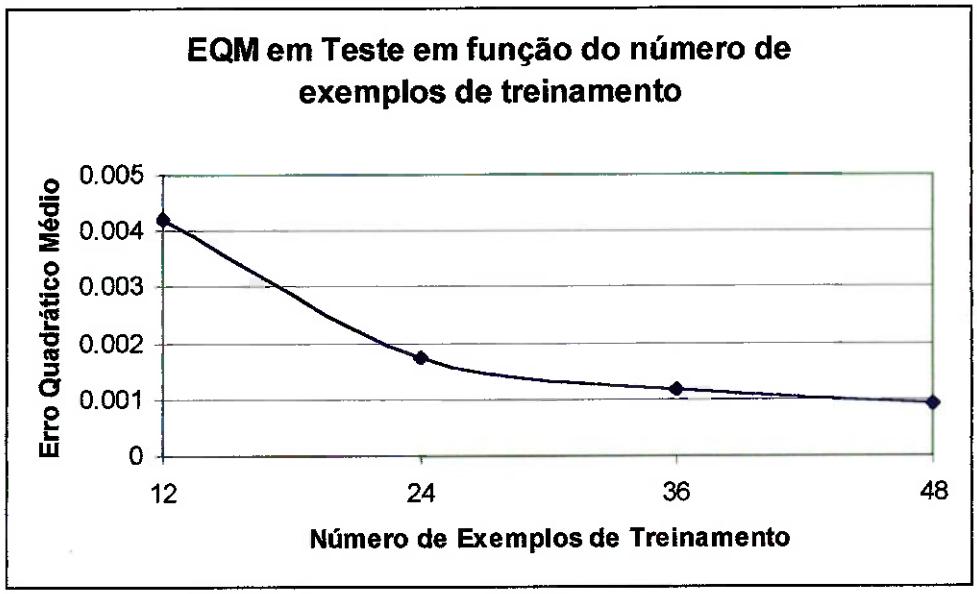


Erro Quadrático Médio em função do número de Iterações (apresentação logarítmica).

Alguns entrevistados forneceram mais de um conjunto de treinamento, o que possibilitou um estudo mais aprofundado do efeito do número de exemplos no treino da RNA.

Parte dos dados foi separada para a realização de testes de desempenho, para verificar se as redes treinadas estavam realmente assimilando (reconhecendo) o padrão de decisão utilizado pelo entrevistado.

O exemplo a seguir foi elaborado com cinco conjuntos de dados (cada um com doze exemplos) fornecidos por um entrevistado da simulação de seleção de parceiros. Destes cinco conjuntos, quatro foram utilizados em treino (12 exemplos, depois 24, depois 36 e depois 48) e os 12 exemplos restantes foram utilizados para teste.



Erro Quadrático Médio em Teste em função do número de exemplos de treinamento.

Verificou-se que redes treinadas com 12, 24, 36 e 48 exemplos convergiram, porém ao serem submetidas a testes, foi possível perceber que quanto maior o número de exemplos em treinamento, menor foi o erro em teste. A rede treinada com 48 exemplos apresentou, para o conjunto de teste, um erro quadrático médio inferior à tolerância exigida em treino, o que indica que a rede se sentiu tão confortável com os dados de teste quanto com os de treino.

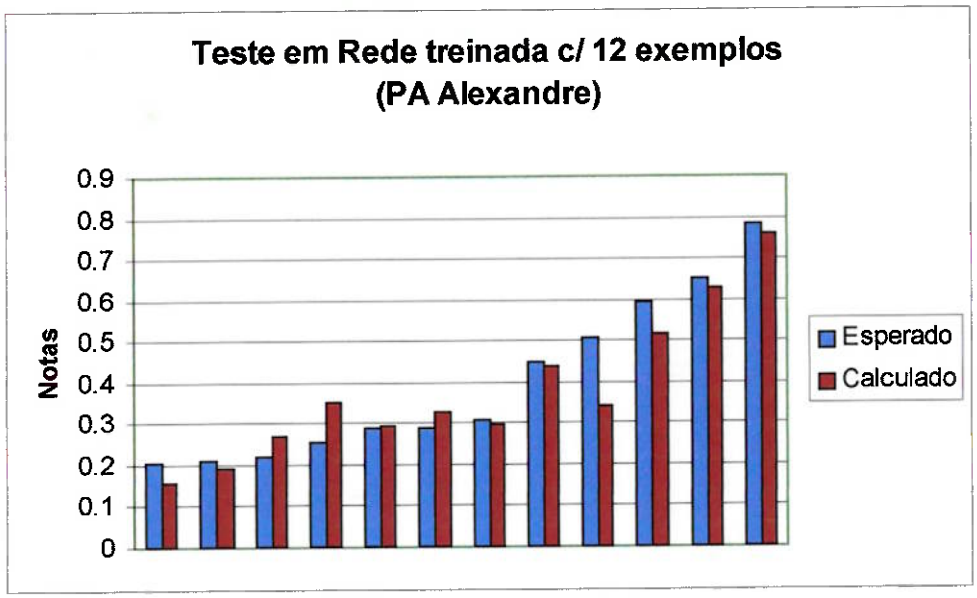
Isso provavelmente significa que, treinada com poucos exemplos (12), a rede tende a reconhecer um padrão sim, mas não o mais genérico possível. Reconhece um padrão mais próximo dos poucos exemplo que lhe foram apresentados, atingindo assim uma capacidade de generalização bem menos ampla do que a do padrão de tomada de decisão adotado pelo entrevistado.

A rede treinada com 48 exemplos conseguiu um erro em teste muito pequeno, inferior à própria tolerância de treinamento, o que significa que esta sim atingiu um padrão de escolha com alta capacidade de generalização (suficiente para acomodar os dados de teste, que nunca foram apresentados à rede).

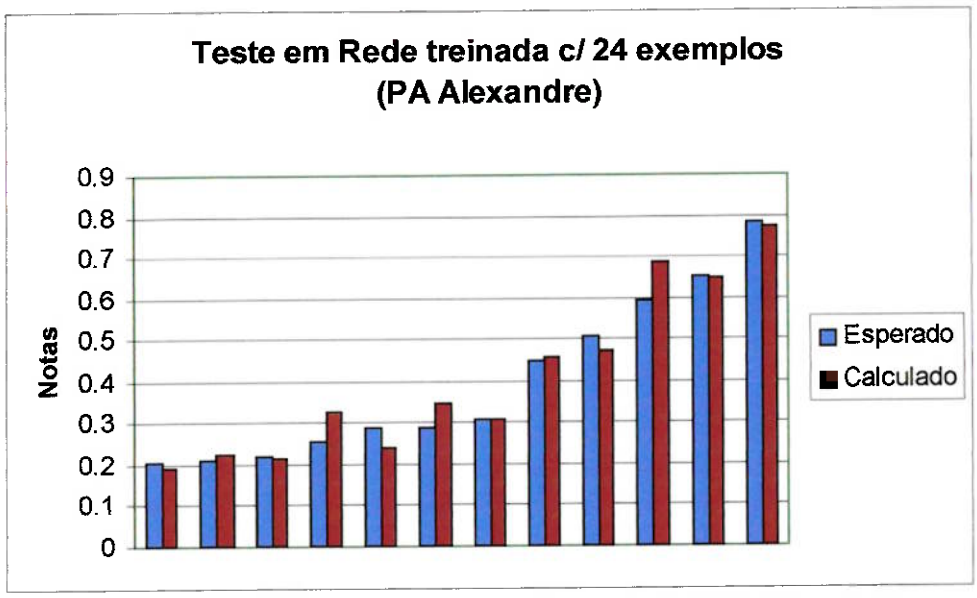
A pequena diferença do erro em teste encontrado pela rede de 36 exemplos do erro da rede treinada com 48 exemplos indica que já ocorre por ali uma saturação. Isso sinaliza que não é necessário um conjunto monstruoso de dados para se atingir um bom treinamento, mas esse número deve estar acima de um valor mínimo (que deve ser determinado para cada situação específica) para que se possa considerar que a rede reconheceu um padrão de escolha com alta capacidade de generalização.

É provável que o método utilizado neste trabalho seja uma abordagem interessante: ir aumentando gradativamente o número de exemplos em treino e ir medindo o erro em teste. A partir do momento em que o erro em teste começar a cair muito lentamente, pode-se considerar que aí começa uma tendência de saturação e o número de exemplos correspondente pode ser considerado um número mínimo de dados para o treinamento em questão.

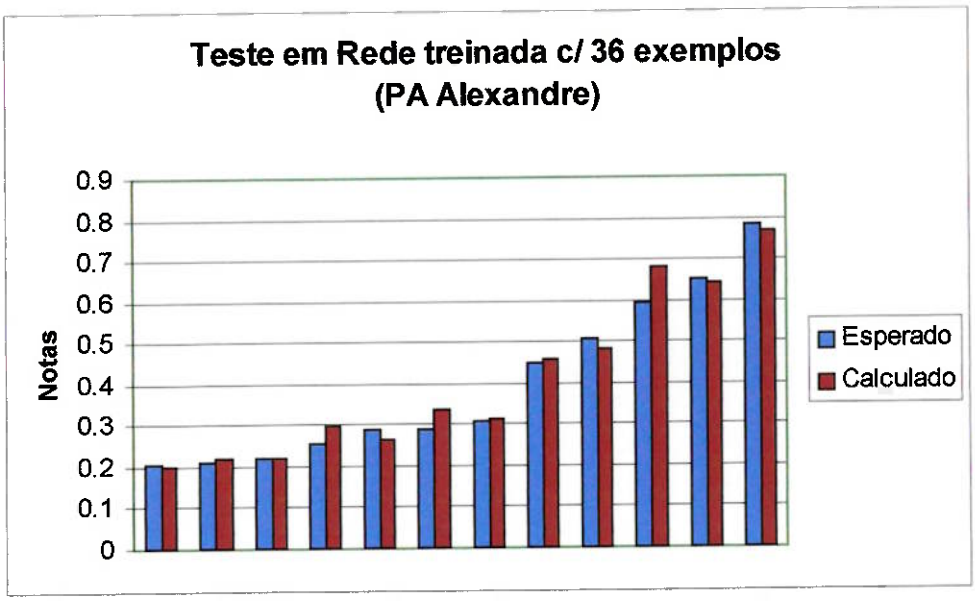
Os gráficos apresentados a seguir comparam os valores (notas para as alternativas) obtidos em teste com os valores conhecidos do conjunto de dados, para um dos entrevistados da simulação de seleção de parceiros.



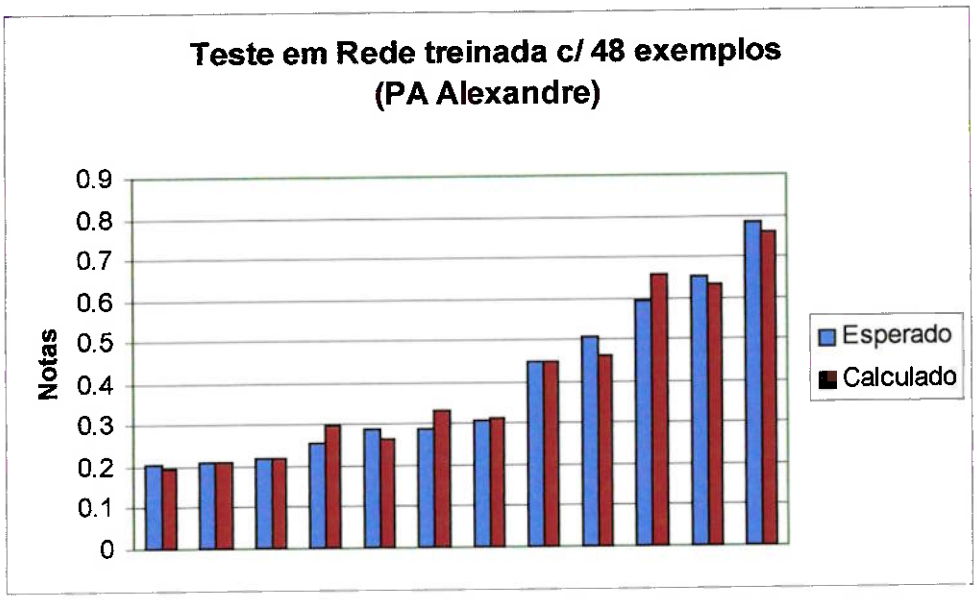
Valores esperados e calculados em teste de uma Rede treinada com 12 exemplos.



Valores esperados e calculados em teste de uma Rede treinada com 24 exemplos.

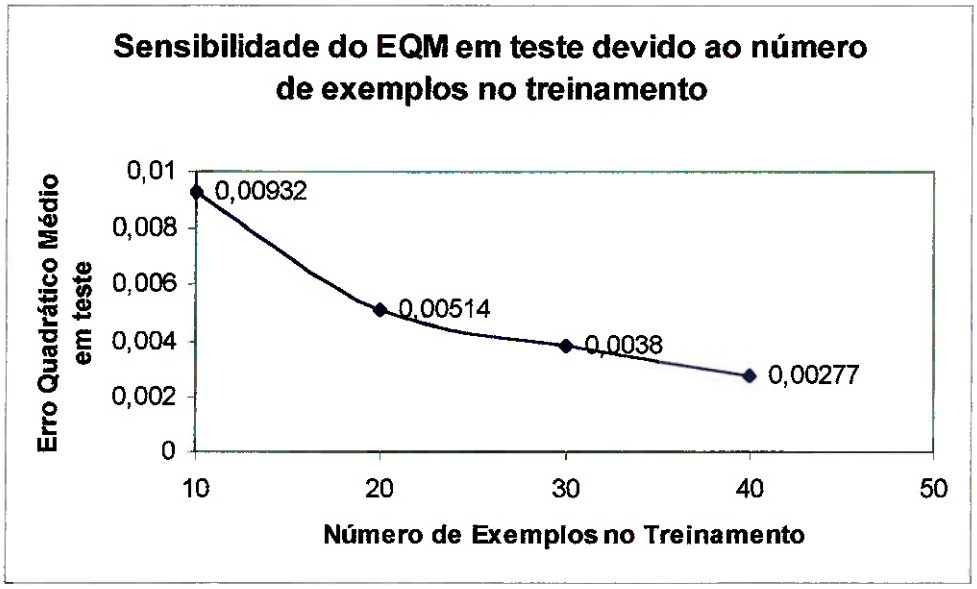


Valores esperados e calculados em teste de uma Rede treinada com 36 exemplos.



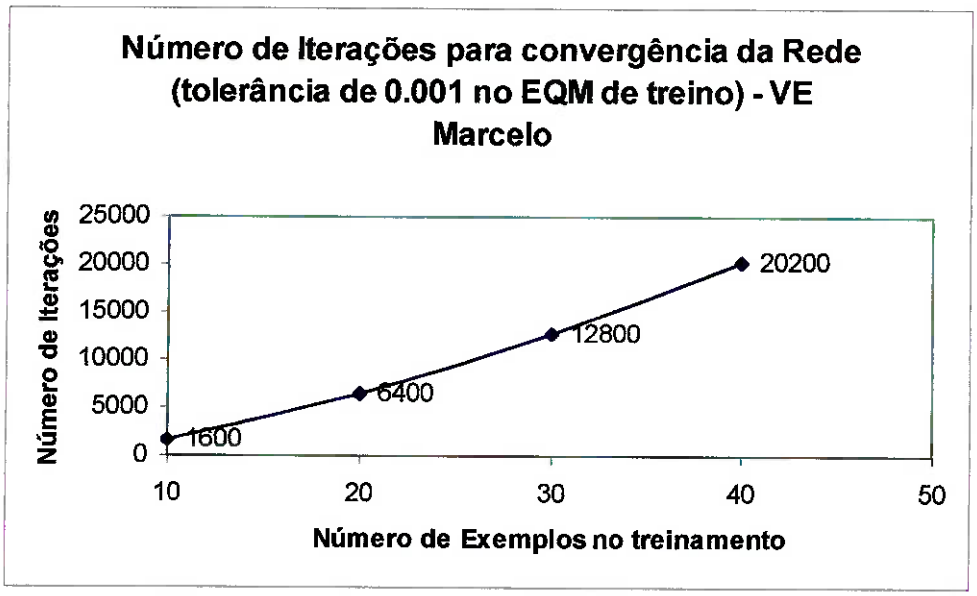
Valores esperados e calculados em teste de uma Rede treinada com 48 exemplos.

Os gráficos a seguir são relativos a uma rede treinada com conjuntos de dados de 10, 20, 30 e 40 exemplos, da simulação de seleção de veículos. Os valores obtidos permitem analisar a influência do número de exemplos em alguns parâmetros. Os comportamentos destas curvas foram comuns em todas as análises.



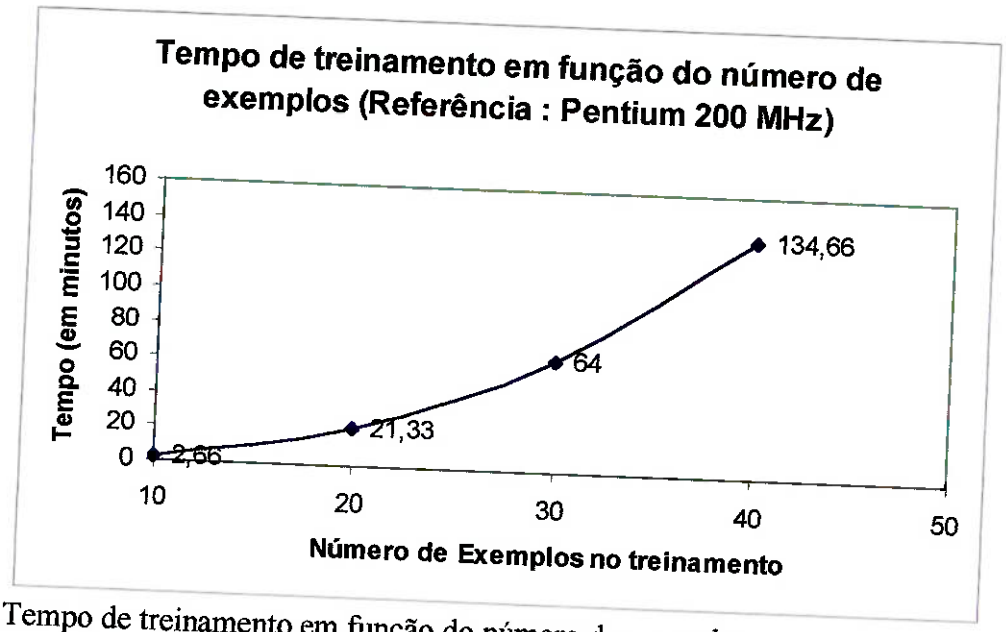
Sensibilidade do EQM em teste devido ao número de exemplos no treinamento.

Neste gráfico, a tendência de saturação não pode ser percebida entre 30 e 40 exemplos, o que também pôde ser verificado em outros casos. No entanto, em todos os casos apareceu uma diferença muito grande entre o erro quadrático médio obtido com 10 exemplos e com 20 exemplos. Sinal de que 10 exemplos (tamanho da maioria dos conjuntos de treinamento fornecidos pelas pesquisas) representam um conjunto de dados insuficiente.



Número de Iterações até atingir a convergência.

Infelizmente, o número de iterações necessário para atingir a convergência cresce rapidamente com o aumento no número de exemplos do conjunto de treinamento. Isso é devido ao fato de que quanto maior o conjunto de treinamento, mais complexo (e mais próximo da realidade) é o padrão que a rede deve reconhecer.



Tempo de treinamento em função do número de exemplos.

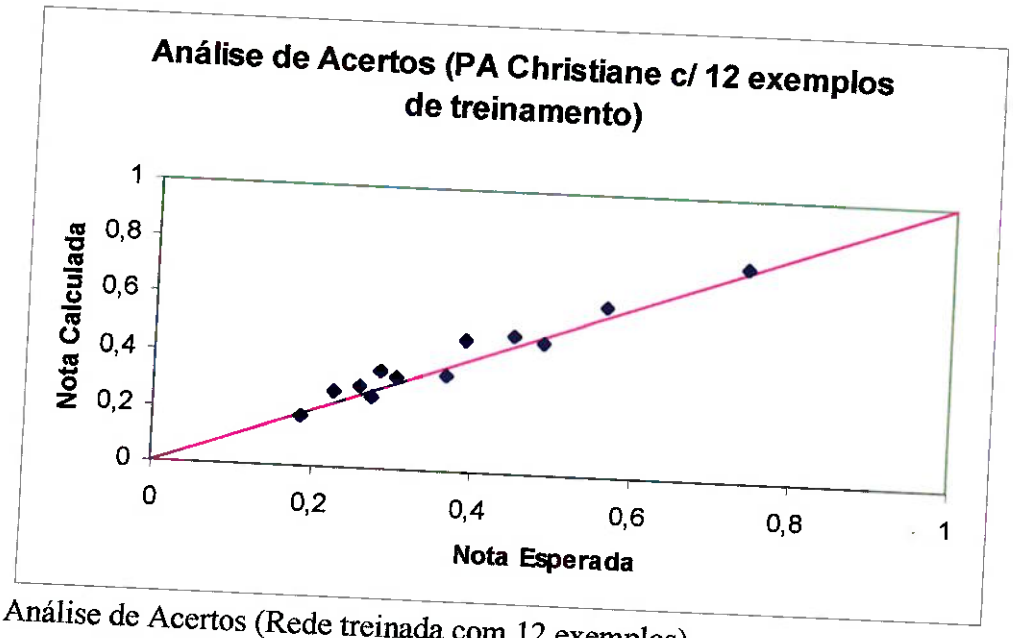
Consequência direta da verificação anterior, o tempo de processamento utilizado pelo treinamento da rede também cresce rapidamente com o aumento do número de exemplos do conjunto de dados.

Conforme pôde ser observado, este crescimento é ainda mais acentuado do que o crescimento no número de iterações. Isso pode ser explicado pelo fato de que, quanto maior o número de exemplos, mais demorado é o processamento de cada iteração (que submete todos os dados do conjunto à rede). Paralelamente, o número de iterações também cresce, o que resulta em taxas de crescimento mais elevadas para o tempo total de processamento.

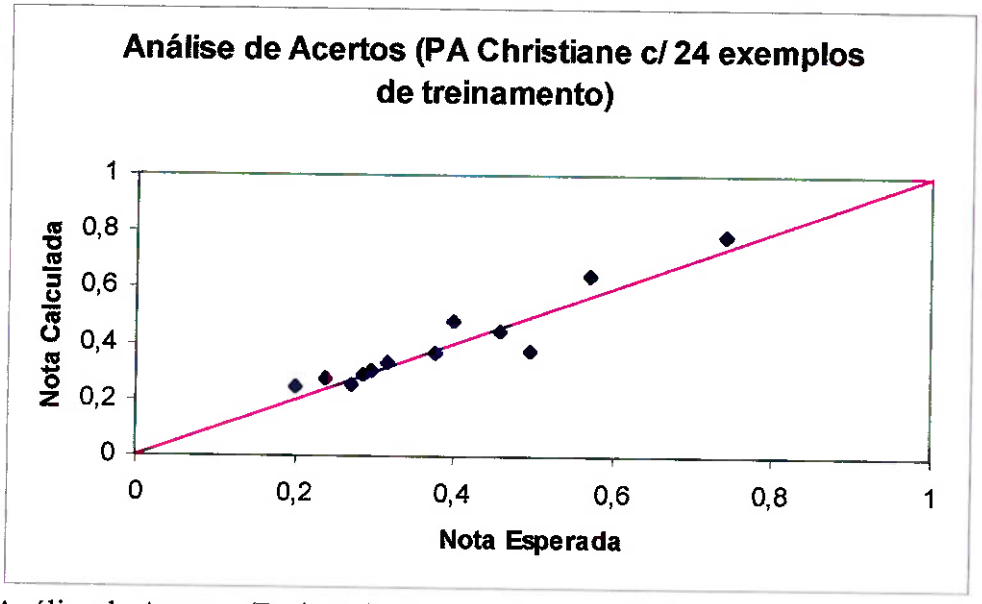
Outra ferramenta utilizada para verificar a eficiência da rede no reconhecimento de padrões de escolha foram os gráficos de análise de acertos,

que também ajudaram a comprovar a importância do número de exemplos em treinamento.

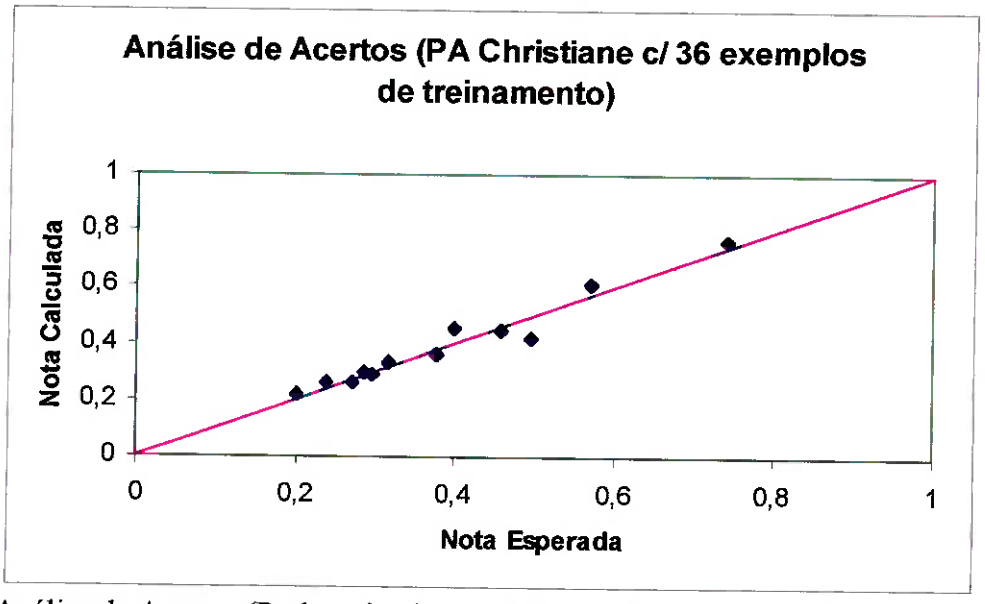
Nestes gráficos, relativos aos resultados obtidos com uma rede treinada com 12, 24, 36 e 48 exemplos da simulação de seleção de parceiros, os pontos têm como abscissas os seus valores (notas das alternativas) esperados, e como ordenadas os seus valores calculados. No caso de acerto total, todos os pontos estariam incluídos na bissetriz do gráfico.



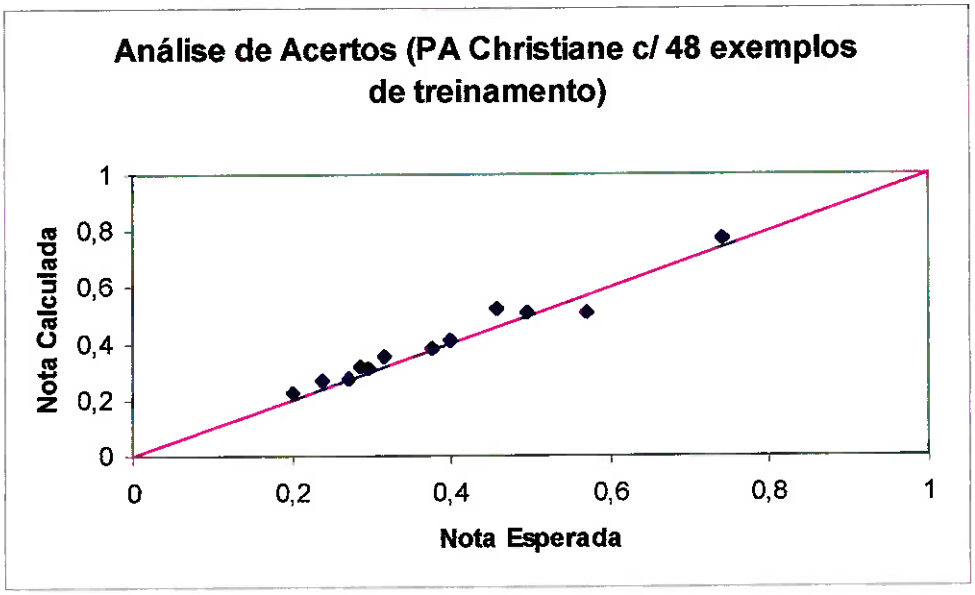
Análise de Acertos (Rede treinada com 12 exemplos).



Análise de Acertos (Rede treinada com 24 exemplos).



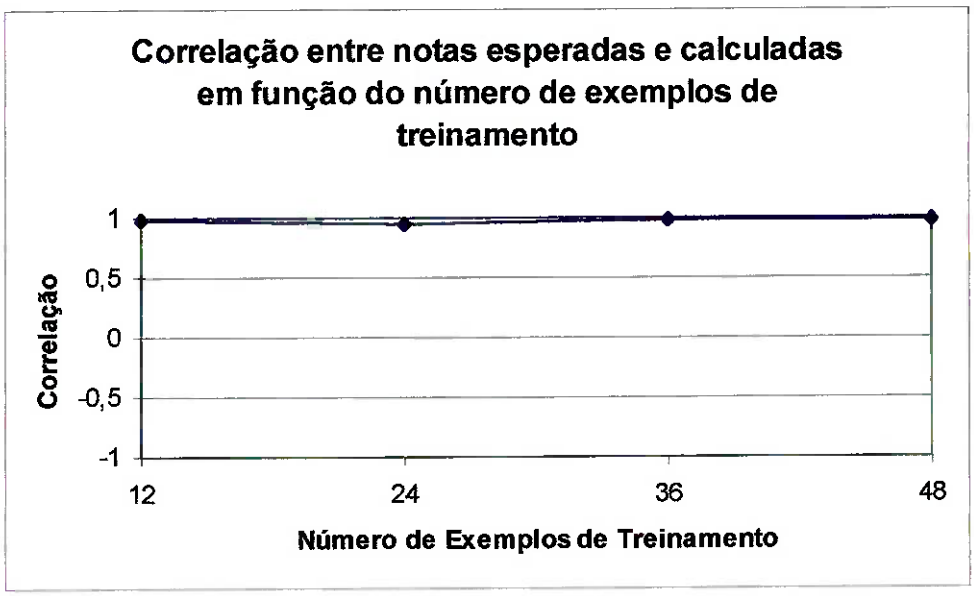
Análise de Acertos (Rede treinada com 36 exemplos).



Análise de Acertos (Rede treinada com 48 exemplos).

O que pode ser verificado a partir da análise destes gráficos é que os pontos relativos ao treinamento com 48 exemplos apresentam uma disposição mais alinhada que os relativos aos outros treinamentos. Além disso, esse alinhamento pode ser bem aproximado pela bissetriz do gráfico, que é a linha que representa a exata compatibilidade entre os valores (notas das alternativas) calculados e os valores esperados.

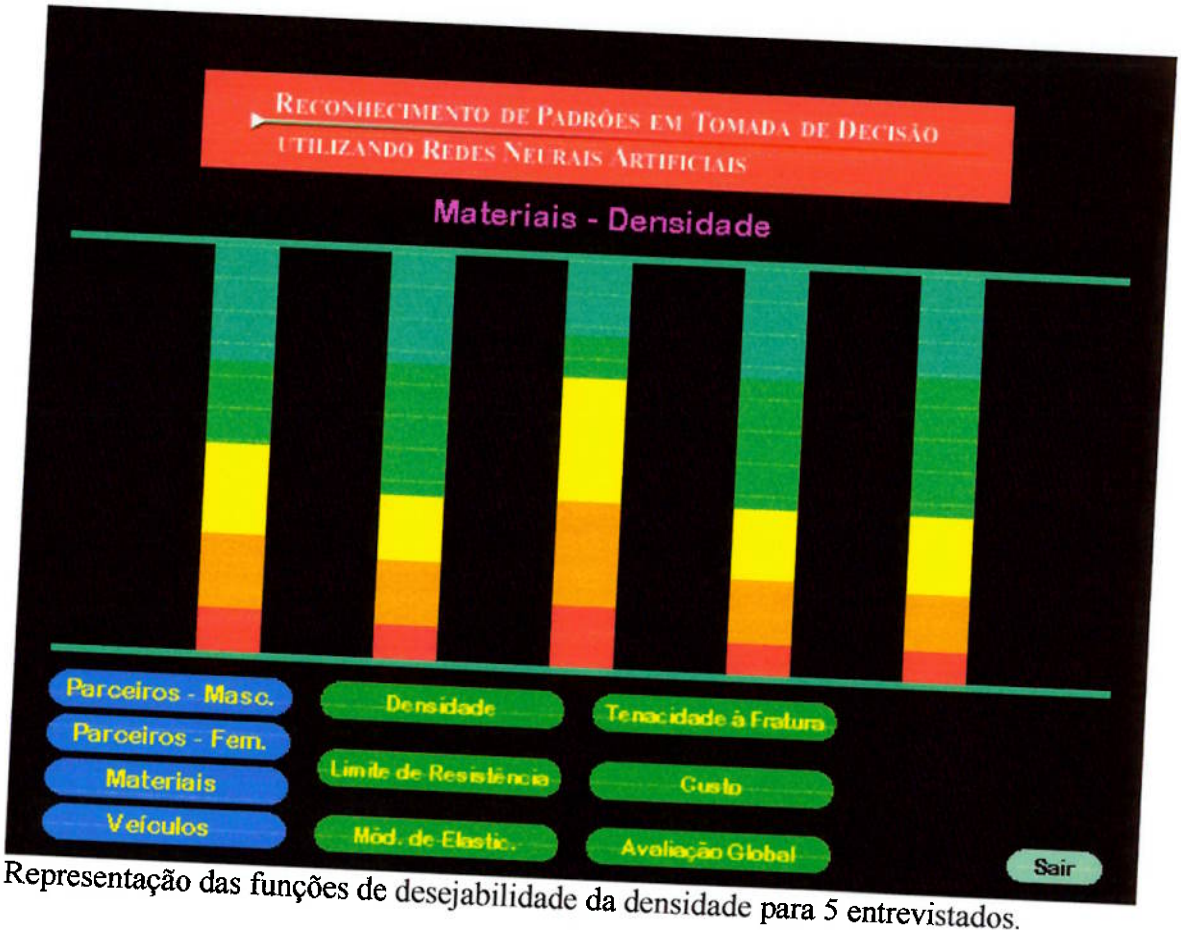
A partir destes dados, é possível obter uma curva de correlação entre valores (notas) esperados e calculados em função do número de exemplos em treinamento.



Correlação entre notas esperadas e calculadas em função do número de exemplos.

Uma correlação de 1,0 indica acordo total, -1,0 indica desacordo total e 0,0 indica indiferença. A curva obtida representa um elevado índice de acordo, o que prova a eficiência da RNA em reconhecer um padrão de decisão.

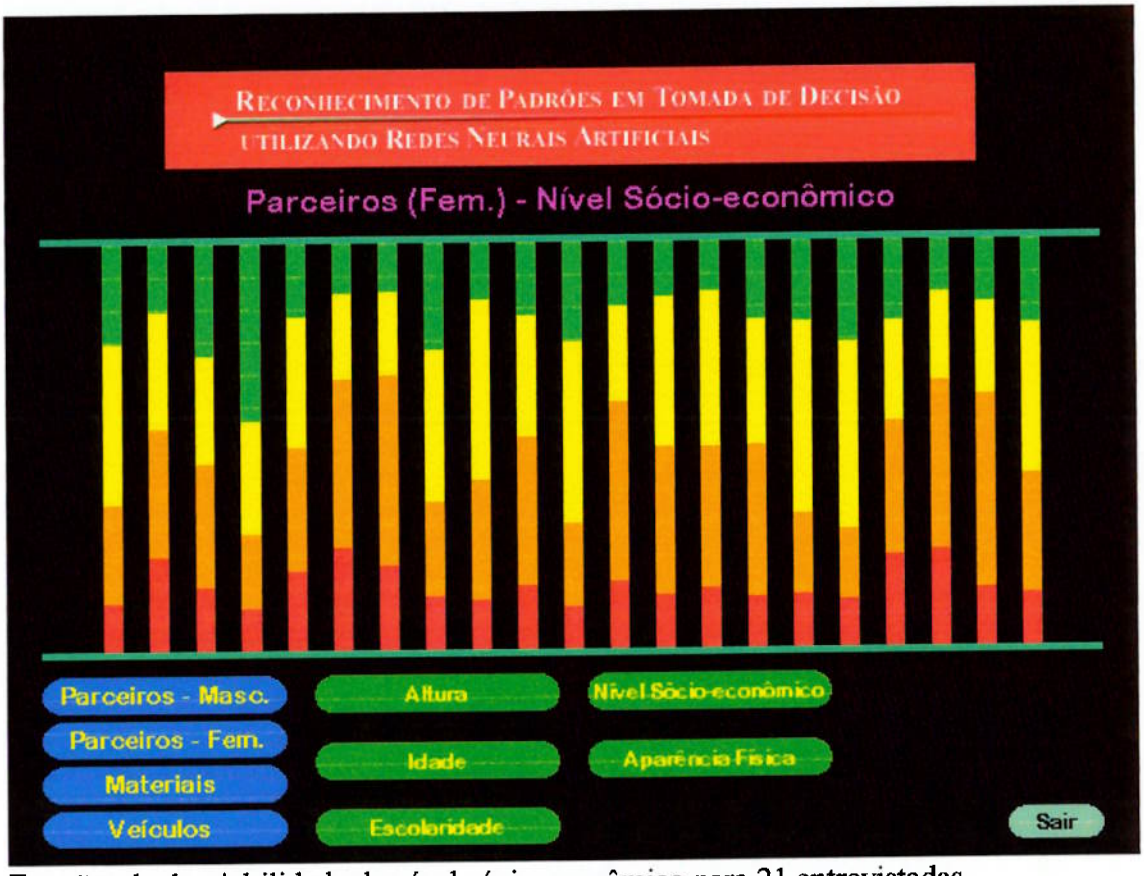
Dos resultados obtidos pelo programa de Pesquisa de Definição das Funções de Desejabilidade (PDFD), foi possível verificar a existência ou não de acordo entre entrevistados quanto às funções de desejabilidade.



Representação das funções de desejabilidade da densidade para 5 entrevistados.

No gráfico anterior, cada coluna representa um entrevistado, e cada cor uma nota (normalizada) referente a um valor de densidade. Pode-se observar que, exceto pelo terceiro entrevistado, houve um elevado grau de acordo entre as funções de desejabilidade de densidade para os outros quatro entrevistados.

No gráfico a seguir, cada coluna representa uma entrevistada, e cada cor uma nota (normalizada) referente a um valor de nível sócio-econômico. Nesse caso, não se pode considerar que existiu acordo entre as funções de desejabilidade de nível sócio-econômico para as entrevistadas.



Funções de desejabilidade do nível sócio-econômico para 21 entrevistadas.

6. CONCLUSÕES

6.1. Peculiaridades da Utilização de RNA

Foi necessário um grande número de testes preliminares antes que se pudesse considerar que o algoritmo de aprendizado da RNA implementada estivesse com um bom desempenho.

Vários "truques" (ajustar as entradas com valores entre 0,2 e 0,8, a inclusão do termo de arrefecimento, etc.) produziram efeitos surpreendentemente decisivos na performance da RNA.

Esses recursos revelam peculiaridades que não são óbvias ou intuitivas para alguém que tenha tido um contato pouco aprofundado com Redes Neurais Artificiais.

Não fosse pela experiência do Prof. Godoy, adquirida ao longo de uma série de trabalhos anteriores em Inteligência Artificial, é bastante provável que a rede estivesse tentando convergir até agora os primeiros conjuntos de treinamento.

O que demonstra que essa aplicação para Redes Neurais Artificiais não é das mais triviais, e portanto não é indicada para servir como um primeiro contato com essa tecnologia.

Algo que deve ser considerado também é que o algoritmo de aprendizado não é um processo determinístico, conforme explicado no capítulo que trata de Redes Neurais Artificiais.

Quando um mesmo conjunto de treinamento era apresentado à rede, esta nem sempre assumia a mesma configuração depois de treinada. Como os pesos iniciais são aleatórios, é possível que a configuração inicial "prenda" a rede num ponto de mínimo local que seja significativamente elevado. Nesse caso, a rede vai parecer ter convergido, e o erro obtido pode ser errôneamente interpretado como indicando uma ausência de padrão de decisão nos dados da simulação.

A solução é treinar a rede várias vezes com o mesmo conjunto de dados, o que diminui a chance de ter uma rede que convergiu para um ponto de elevado mínimo local.

6.2. Independência quanto às Funções de Desejabilidade

Uma importante consideração a respeito do mecanismo de reconhecimento de padrões parte da comparação entre funções de desejabilidade de diferentes entrevistados.

Conforme verificado no fim no capítulo anterior, pode existir ou não acordo entre as funções de desejabilidade de diferentes entrevistados (existiu no caso da densidade, e não existiu no caso do nível sócio-econômico).

É importante que se note, no entanto, que desacordo entre funções de desejabilidade e desacordo entre seleção de alternativas viáveis não implica necessariamente em desacordo quanto ao padrão de escolha.

Esse foi um resultado bastante comum, especialmente na simulação de seleção de parceiros. Assim como a função de desejabilidade do nível sócio-econômico, outros critérios apresentaram desacordo.

No entanto, a RNA conseguiu detectar padrões de decisão comuns mesmo entre entrevistados discordantes quanto às funções de desejabilidade de alguns critérios. Dados de um entrevistado funcionavam bem quando testados em rede treinada com dados do outro.

Esse mesmo fato pôde ser verificado quando a RNA detectou um afinado padrão médio para o grupo de entrevistados (tanto masculinos quanto femininos, mas cada um em seu grupo) da simulação de seleção de parceiros.

Isso se deve ao fato de que é possível que entrevistados possam discordar quanto a qual é o valor mais importante para um critério, mas concordar quanto à importância deste critério em relação aos demais.

6.3. Efeito da Pré-concepção de Alternativas

Ao analisar os resultados das pesquisas da simulação de seleção de materiais, surgiu uma interessante descoberta.

A simulação de seleção de materiais é possivelmente a única das três que admite uma clara distinção, no grupo dos entrevistados, entre especialistas (professores da área) e "leigos" (alunos de graduação, com conhecimentos básicos de propriedades mecânicas dos materiais).

Eram assim esperados padrões mais bem definidos entre os especialistas do que entre os "leigos". Porém isso não ocorreu. Alguns "leigos" apresentaram, inclusive, padrões mais bem definidos do que alguns especialistas.

Isso se deve, provavelmente, ao fato de que embora os materiais tenham sido apresentados como hipotéticos, os especialistas conseguiram identificar algumas descrições como sendo próximas às de certos materiais, e assim levar em consideração, ainda que subliminarmente, outros critérios que não estavam em discussão (e que não deveriam ser considerados). Significa que eles deram força para pré-concepções, e isso se refletiu como uma fonte de erro dentro do padrão, já que a partir daí eles passaram a rotular as alternativas (materiais) e a fazer julgamentos a partir destes rótulos.

Enquanto isso os "leigos", por não conseguirem identificar os materiais, foram forçados a centrar a decisão somente nos critérios apresentados. Tomaram, assim, decisões mais "criteriosas", o que levou aos padrões mais bem definidos.

6.4. Coerência no Processo de Tomada de Decisão

Para uma correta interpretação dos resultados, deve ser levada em consideração uma observação pertinente, ainda que óbvia: não tem a menor importância para a Rede Neural a validade da simulação de uma escolha. A escolha pode ser totalmente "errada" (preferir carros mais caros, por exemplo) que ainda assim não há o menor problema para a rede, desde que essa escolha errada siga a uma determinada lógica (errada) possível de ser operacionalizada apenas com os critérios apresentados. É claro que, neste caso, a rede reconhecerá um padrão bem definido, mas errado.

Por outro lado, um padrão de escolha absolutamente perfeito pode não ser reconhecido por uma RNA se essa escolha se basear numa lógica que extrapola os critérios apresentados.

Resumindo, para conseguir obter um bom padrão, não é necessário ser correto, e sim ser coerente.

6.5. Critérios Quantitativos e Qualitativos

Houve um certo cuidado inicial com relação aos critérios de avaliação subjetivos, aqui denominados "qualitativos" (como aparência, por exemplo). Ao longo de todo o trabalho, no entanto, todos os critérios foram tratados exatamente

da mesma maneira, se submetendo à escala subjetiva apontada pelo Método de Saaty.

Se houve a desconfiança de que as seleções baseadas em critérios mais numéricos, "quantitativos", teriam maior aplicabilidade no tomador de decisão, ela caiu por terra.

O Método de Saaty colocou todos os critérios no mesmo patamar de subjetividade (no caso, de total subjetividade). Que na prática é o que importa no momento de decisão (o quanto um valor afeta o indivíduo que fará uma escolha, e não o valor em si).

A simulação baseada estritamente em critérios "quantitativos", a de seleção de materiais, talvez até pelas razões já descritas, foi a que obteve os padrões menos definidos.

E ainda que todas as simulações tenham sido sucedidas em possibilitar o reconhecimento de um padrão, a simulação de parceiros foi a que se revelou mais adequada para ser emulada por um tomador de decisão (apresentou os padrões mais definidos). Justo ela, que só possuía critérios ditos "subjetivos".

6.6. Importância do Número de Exemplos em Treinamento

Os resultados obtidos nos testes de desempenho deixaram clara a importância do número de exemplos em treinamento. A análise da influência desse número no erro quadrático médio e na correlação entre notas calculadas e

esperadas sugere que, para o grau de complexidade das simulações realizadas, é necessário um conjunto de treinamento com pelo menos 40 exemplos.

Infelizmente, a maior parte dos entrevistados forneceu apenas 8, 10 ou 12 exemplos. Nesses casos, cada rede treinada com esses dados convergiu para um padrão, na maioria das vezes até muito bem definido. Que embora apresentasse um ajuste muito preciso com os dados de treino, não revelou esse mesmo ajuste com os dados de teste. O que faz com que esse padrão não possa ser considerado representativo da "lógica" de escolha do entrevistado.

Já as redes treinadas com conjuntos maiores conseguiram "acomodar" o conjunto de teste tão bem quanto o conjunto de treinamento. Demandaram treinamentos mais demorados e reconheceram padrões mais complexos. Mas o fato é que apenas estes padrões atingiram a capacidade de generalização necessária para que pudessem ser considerados fiéis emuladores das "lógicas" de escolha dos entrevistados para aquelas situações específicas.

Essa relevância do número exemplos em treinamento não deve ser atribuída somente à metodologia de reconhecimento baseada em Redes Neurais. O que ocorre é que um conjunto de treinamento pequeno contém um número de exemplos que não consegue ilustrar completamente todas as sutilezas e regras que caracterizam e refletem uma lógica de decisão. Independente de qual a metodologia adotada, não é possível reconhecer um padrão de decisão numa situação de insuficiência de dados.

O problema é que conjuntos de treinamento mais numerosos são resultados de processos de pesquisa longos e extenuantes. Em algumas aplicações, isso seria imediatamente traduzido como maiores custos.

No entanto, em algumas aplicações práticas de tomadores de decisão, isso pode não representar problema algum. Caso se possa implementar um sistema de monitoramento contínuo de uma situação real e cotidiana de tomada de decisão, esse sistema pode armazenar as decisões tomadas (saídas), assim como as informações que levaram a elas (entradas) e alimentar uma RNA. Dessa forma, o conjunto de treinamento iria crescer naturalmente.

Ainda nesse caso, a partir de certo momento a rede já poderia ir sendo testada com as informações novas (antes de serem incorporadas ao conjunto de treinamento). Quando o padrão adquirido tiver atingido um determinado desempenho mínimo, pode-se adotar uma fase de "controle assistido": a rede, alimentada apenas com as entradas, sugere uma decisão, que pode ser acatada ou rejeitada pelo agente da decisão (a pessoa responsável pela escolha), e o resultado volta como mais uma informação para o conjunto de treino.

Quando o agente da decisão perceber que está acatando praticamente todas as sugestões do tomador de decisão, é sinal de que o sistema adquiriu certa confiabilidade.

6.7. Restrições de Confiabilidade

Não se deve considerar, em hipótese alguma, que a confiabilidade do emulador de tomada de decisão é irrestrita. A própria aplicação de RNA já traz a característica intrínseca de possuir elevada capacidade de interpolação, e não de extrapolação. O que significa que para situações similares àquelas já apresentadas à rede, os resultados serão muito próximos da "lógica" de decisão do(s) agente(s) estudado(s). Mas nada garante que o padrão reconhecido pela rede possa ser aplicado numa situação inesperada, diferente de tudo o que já foi apresentado durante o seu treinamento.

Não se pode esperar da rede uma "consciência" na tomada de decisão. Nem se pode admitir que ela passou a raciocinar em ressonância com o agente estudado. A verdade, nua e crua, é que a RNA treinada não passa de um sofisticado modelo matemático, algo como um "aproximador de funções" completamente cego, que não compreende (nem poderia compreender) o significado dos valores numéricos que manipula com tanta habilidade.

Isso deve ser levado em consideração em aplicações de alto risco (controle de tráfego aéreo, supervisão de usinas nucleares, etc.), situações onde em geral mesmo a decisão humana necessita, às vezes, ser referendada por terceiros ou ser submetida aos procedimentos de segurança.

Sempre que for avaliada a viabilidade deste tipo de emulador de tomada de decisão em aplicações de alto risco, este deve ser considerado mais uma ferramenta de sugestão de decisão do que de tomada de decisão.

6.8. Número de Critérios

Um cuidadoso treinamento, envolvendo uma bem elaborada estratégia de pesquisa com grupos de especialistas, pode resultar num padrão extremamente representativo e bem comportado. Um emulador de tomada de decisão baseado neste padrão traria a possibilidade de automatizar completamente tarefas rotineiras de controle. A capacidade das RNA em lidar com um grande número de informações deixa-as em condição de vantagem em relação às pessoas em situações envolvendo grande número de fatores a serem considerados numa decisão.

Em geral, em situações deste tipo as pessoas tendem a adotar metodologias de classificação de critérios como a curva ABC, onde os critérios considerados mais importantes (A) são destacados dos intermediários (B) e dos menos importantes (C). No fundo, trata-se apenas de uma estratégia que elimina a influência dos fatores menos importantes, enaltecendo a observação clássica de que a mente humana só consegue considerar 7±2 fatores simultaneamente (SAATY, 1974).

A RNA leva em consideração qualquer número de fatores, por mais insignificantes que sejam os efeitos de alguns deles. Vale lembrar que o efeito acumulado de uma porção de fatores pouco relevantes pode superar o efeito de um fator mais relevante.

Cabe aqui a pergunta de como a influência de um grande número de fatores pode ser assimilada partindo da análise da decisão humana, que automaticamente ignora as menos importantes devido à sua natural limitação. Isso pode ser resolvido organizando uma estratégia de pesquisa que combine (e não divida) os diversos fatores em pequenos grupos (por exemplo de cinco elementos como neste trabalho) e submeteria todos a pesquisa. Incluindo um critério de referência em dois conjuntos diferentes permite estabelecer um termo de comparação comum aos dois conjuntos, e assim reuní-los novamente.

Dessa forma, o padrão poderá ser considerado mais criterioso (obedece coerentemente a um maior número de critérios) do que qualquer agente humano. Embora isso não necessariamente leve a uma decisão mais acertada do ponto de vista do agente, que devido à sua limitação de número de fatores ficará na dúvida entre desconfiar do emulador de tomada de decisão (que pode ter se deparado com uma situação extraordinária e estar errado) ou acatar a decisão do emulador e desconfiar da sua própria capacidade de julgamento.

Pode também ser desenvolvido um mecanismo de avaliação, por parte da RNA, da capacidade de julgamento das pessoas, com finalidade de seleção ou treinamento.

6.9. Considerações Finais

Os resultados comprovaram a expectativa inicial de que é possível emular, com razoável confiabilidade, um processo de tomada de decisão particular e específico, bem definido por um limitado conjunto de alternativas e critérios.

O processo de decisão humano é frequentemente tratado de forma mítica. Um sem número de expressões populares e lugares comuns tecem a idéia de que em várias circunstâncias o nosso "lado racional" é posto de lado, principalmente quando a escolha envolve aspectos subjetivos.

Essa idéia é inclusive bastante confortável, já que ergue uma barreira aparentemente intransponível entre o homem e a máquina, já que esta não opera com emoções. Sinaliza, com certo alívio, que justamente por isso a máquina jamais superará o homem. Deixando de lado a preocupação, até certo ponto infantil, da concorrência com as máquinas, é interessante que se passe a encarar de forma menos mítica e mais objetiva esse assunto.

Na nossa sociedade, é gigantesco o número de aplicações para um tomador de decisão, desde que ele seja confiável. Do controle de tráfego ao planejamento industrial, significativa parte de nossos recursos são diariamente desperdiçados por causa de decisões atrasadas, baseadas em lógicas ultrapassadas ou simplesmente erradas.

Em controle, tradicionalmente o sistema a ser controlado é exaustivamente estudado e modelado, para que se possa conhecer minuciosamente o seu comportamento dinâmico. Em alguns casos, esses sistemas são complexos demais, difíceis de serem modelados ou até mesmo completamente desconhecidos quanto ao seu comportamento. Nesses casos, por que não ignorar o sistema e encarar o seu operador humano como sendo o objeto de controle ? Suas ações (saídas) são baseadas em decisões feitas a partir de um conjunto de informações (entradas).

A inteligência do homem dispõe das mais avançadas ferramentas de adaptação e aprendizado. Nada mais plausível, portanto, que ele adquira domínio sobre o controle de um sistema ao qual ele nunca foi apresentado, apenas baseando-se na sua capacidade de observação e dedução, aliada à sua experiência prática. E essa é a razão pela qual operações de controle tão triviais ao homem (como dirigir um carro) sejam tão absurdamente difíceis de reproduzir num ambiente artificial.

Tendo em vista esta capacidade, é fácil defender que uma excelente estratégia de modelagem e projeto, especialmente em controle, consiste em delegar ao homem a tarefa de se familiarizar com um sistema, compreender o seu comportamento dinâmico, dominar o seu controle, e aí então "extrair" este conhecimento adquirido transmitindo-o, através de uma bem planejada pesquisa, a um sistema de reconhecimento de padrões, que pode assim aprender e executar a tarefa de controle.

Como sempre, dedicando à máquina a parte mais laboriosa e repetitiva, cabendo ao homem a tarefa mais "divertida", exercitando a sua curiosidade e capacidade de aprendizado. Mas para isso é fundamental que a máquina possa aprender com o homem, reconhecendo os padrões contidos nas suas decisões.

A elevada correlação entre valores esperados (conhecidos) e valores calculados em redes treinadas com grande número de exemplos tornou impossível identificar qual conjunto de dados pertencia a qual fonte. Em outras palavras, não foi possível, nestes casos, distinguir a escolha do homem da escolha da "máquina". Sinal de que esta já deve ter aprendido alguma coisa.

ANEXO – ROTINAS CIENTÍFICAS DOS PROGRAMAS

Foram relacionadas apenas as rotinas científicas dos programas, ficando de fora todas as inúmeras rotinas de apresentação de dados em tela, de entrada de dados, de manipulação de arquivos, etc., que só fariam dobrar o número de páginas deste trabalho, sem acrescentar nada de relevante na descrição da metodologia implementada.

Pesquisa de Calibração do Método de Saaty (PCMS)

```
on montaMatriz i

  global matrizS,nCrit,limCrit,seqCities,limCont,¬

         Resp1,Resp2,Resp3,Resp4,Resp5,Mat


  set matrizS to ""

  put (getAt (limCrit,i)) into ordem
  set Mat to abreMatriz (ordem)

  case i of
    1 : set seqV to seqCities
  end case

  case i of
    1 : set RespV to Resp1
    2 : set RespV to Resp2
    3 : set RespV to Resp3
    4 : set RespV to Resp4
    5 : set RespV to Resp5
  end case

  repeat with j = 1 to getAt (limCont,i)
    put getAt (RespV,j) into nor
    put nor * 8 into expand
    if expand > 0 then
      put expand + 1 into expand
    else
      put expand - 1 into expand
    end if
    put expand / abs (expand) into sinal
    put abs (expand) into expand

    put getAt (seqV,(j-1)*2+1) into esq
    put getAt (seqV,(j-1)*2+2) into dir

    if sinal = 1 then
      insere (Mat,ordem,dir,esq,expand)
      insere (Mat,ordem,esq,dir,1/expand)
    else
      insere (Mat,ordem,esq,dir,expand)
      insere (Mat,ordem,dir,esq,1/expand)
    end if
  end repeat

end montaMatriz
```

```
on abreMatriz ordem

  put ordem into n

  set Mat to [1]
  repeat with u1 = 1 to n-1
    repeat with u2 = 1 to n
      add Mat,0
    end repeat
    add Mat,1
  end repeat

  return Mat

end abreMatriz



on insere Mat,ordem,x,y,w

  setAt (Mat,ordem*(x-1)+y,w)

end insere



on le Mat,ordem,x,y

  return getAt (Mat,ordem*(x-1)+y)

end le


on escMat Mat,n,x,y,w

  setAt (Mat,n*(x-1)+y,w)

end escMat



on leMat Mat,n,x,y

  return getAt (Mat,n*(x-1)+y)

end leMat



on norm vet

  put 0 into somat
  repeat with i = 1 to count(vet)
```

```
    set somat = somat + getAt (vet,i)
  end repeat

  repeat with i = 1 to count(vet)
    setAt (vet,i,getAt(vet,i)*1.0/somat)
  end repeat

  return vet

end norm



on media vet

  put 0 into somat
  repeat with i = 1 to count(vet)
    set somat = somat + getAt (vet,i)
  end repeat

  set aux = somat * 1.0 / count (vet)

  return aux

end media



on Saaty Mat

  set n = integer (sqrt(count(Mat)))

  repeat with j = 1 to n
    set vetAux = []
    repeat with i = 1 to n
      add vetAux,leMat (Mat,n,i,j)
    end repeat
    set vetAux = norm (vetAux)
    repeat with i = 1 to n
      escMat (Mat,n,i,j,getAt (vetAux,i))
    end repeat
  end repeat

  set escala = []
  repeat with i = 1 to n
    set vetAux = []
    repeat with j = 1 to n
      add vetAux,leMat (Mat,n,i,j)
    end repeat
    add escala, media (vetAux)
  end repeat

  return norm (escala)

end Saaty
```

```
on fit escX

  global t,em,dist,tmin,tmax,nIte,escalaF,emf

  set OK = 1
  set crPar = 0.0001

  set errant = em

  set nIte = nIte + 1

  set errmin = 1
  repeat with i = 0 to 20
    set aux = i*1.0*(tmax-tmin)/20 + tmin
    set em = mmq (eleva (aux))
    if em < errmin then

      set errmin = em
      set t = aux
    end if
  end repeat

  set em = errmin
  put abs(em-errant)/em into esterr


  if (nIte <= 15) then
    set tmin = 1.0/2*(t-tmin) + tmin
    set tmax = 1.0/2*(tmax-t) + t
    fit escX
  else
    put em into emf
    put 0 into OK
  end if


  return eleva (t)

end fit



on eleva ex

  global escalaP

  set escT to []
  repeat with i = 1 to 5
    put getAt (escalaP,i) into bas
    add escT,power (bas,ex)
  end repeat

  return norm (escT)

end eleva
```

```
on mmq escE

  set acum = 0
  repeat with i = 1 to 5
    set dif = getAt (dist,i) - getAt (escE,i)
    set acum = acum + power (dif,2)
  end repeat

  return acum

end mmq



on errPerc escD

  global dist

  set the floatPrecision to 6

  set errAc = 0
  repeat with i = 1 to 5
    set errAc = errAc + abs(getAt(dist,i) - getAt (escD,i)) / ¬
                getAt (dist,i)
  end repeat

  set the floatPrecision to 2

  return errAc / 5

end errPerc



on calc

  global Mat,t,em,dist,escalaP,emp

  set the floatPrecision to 8

  montaMatriz (1)

  set escalaP = Saaty (Mat)

  set escalaF = fit (escalaP)



  set the floatPrecision to 2

  set emp = errPerc (escalaF) * 100.0

end calc
```

```
on prepCalc

  global nIte,tmin,tmax,em

  set nIte = 0
  set tmin = 0.2
  set tmax = 3
  set em   = 1

  calc

end prepCalc
```

Pesquisa de Definição das Funções de Desejabilidade (PDFD)

```
on inicializa

  global nCrit,limCrit,Resp1,Resp2,Resp3,Resp4,Resp5,fLoad

  set Resp1 to []
  put getAt (limCrit,1) into ncb
  put ncb * (ncb-1) / 2 into lim
  repeat with i = 1 to lim
    add Resp1,0
  end repeat

  set Resp2 to []
  put getAt (limCrit,2) into ncb
  put ncb * (ncb-1) / 2 into lim
  repeat with i = 1 to lim
    add Resp2,0
  end repeat

  set Resp3 to []
  put getAt (limCrit,3) into ncb
  put ncb * (ncb-1) / 2 into lim
  repeat with i = 1 to lim
    add Resp3,0
  end repeat

  set Resp4 to []
  put getAt (limCrit,4) into ncb
  put ncb * (ncb-1) / 2 into lim
  repeat with i = 1 to lim
    add Resp4,0
  end repeat

  set Resp5 to []
  put getAt (limCrit,5) into ncb
  put ncb * (ncb-1) / 2 into lim
  repeat with i = 1 to lim
    add Resp5,0
  end repeat

  if fLoad = 0 then carrega
  put fLoad + 1 into fLoad


end inicializa


on montaMatriz i

  global matrizS,nCrit,limCrit,seqApar,seqCons,seqCapC,¬
         seqPrec,seqManu,limCont,Resp1,Resp2,Resp3,Resp4,¬
         Resp5,Mat
```

```
  set matrizS to ""

  put (getAt (limCrit,i)) into ordem
  set Mat to abreMatriz (ordem)

  case i of
    1 : set seqV to seqApar
    2 : set seqV to seqCons
    3 : set seqV to seqCapC
    4 : set seqV to seqPrec
    5 : set seqV to seqManu
  end case

  case i of
    1 : set RespV to Resp1
    2 : set RespV to Resp2
    3 : set RespV to Resp3
    4 : set RespV to Resp4
    5 : set RespV to Resp5
  end case

  repeat with j = 1 to getAt (limCont,i)
    put getAt (RespV,j) into nor
    put nor * 8 into expand
    if expand > 0 then
      put expand + 1 into expand
    else
      put expand - 1 into expand
    end if
    put expand / abs (expand) into sinal
    put abs (expand) into expand

    put getAt (seqV,(j-1)*2+1) into esq
    put getAt (seqV,(j-1)*2+2) into dir

    if sinal = 1 then
      insere (Mat,ordem,dir,esq,expand)
      insere (Mat,ordem,esq,dir,1/expand)
    else
      insere (Mat,ordem,esq,dir,expand)
      insere (Mat,ordem,dir,esq,1/expand)
    end if
  end repeat

end montaMatriz



on abreMatriz ordem

  put ordem into n

  set Mat to [1]
  repeat with u1 = 1 to n-1
    repeat with u2 = 1 to n
      add Mat,0
    end repeat
```

```
    add Mat,1
  end repeat

  return Mat

end abreMatriz



on insere Mat,ordem,x,y,w

  setAt (Mat,ordem*(x-1)+y,w)

end insere



on le Mat,ordem,x,y

  return getAt (Mat,ordem*(x-1)+y)

end le



on escMat Mat,n,x,y,w

  setAt (Mat,n*(x-1)+y,w)

end escMat



on leMat Mat,n,x,y

  return getAt (Mat,n*(x-1)+y)

end leMat



on norm vet

  put 0 into somat
  repeat with i = 1 to count(vet)
    set somat = somat + getAt (vet,i)
  end repeat

  repeat with i = 1 to count(vet)
    setAt (vet,i,getAt(vet,i)*1.0/somat)
  end repeat

  return vet

end norm
```

```
on media vet

  put 0 into somat
  repeat with i = 1 to count(vet)
    set somat = somat + getAt (vet,i)
  end repeat

  set aux = somat * 1.0 / count (vet)

  return aux

end media



on Saaty Mat

  set n = integer (sqrt(count(Mat)))

  repeat with j = 1 to n
    set vetAux = []
    repeat with i = 1 to n
      add vetAux,leMat (Mat,n,i,j)
    end repeat
    set vetAux = norm (vetAux)
    repeat with i = 1 to n
      escMat (Mat,n,i,j,getAt (vetAux,i))
    end repeat
  end repeat

  set escala = []
  repeat with i = 1 to n
    set vetAux = []
    repeat with j = 1 to n
      add vetAux,leMat (Mat,n,i,j)
    end repeat
    add escala, media (vetAux)
  end repeat

  return norm (escala)

end Saaty
```

Pesquisa de Escolha de Alternativas Viáveis (PEAV)

```
on inicializa

  global nCrit,limCrit,Resp1,Resp2,Resp3,Resp4,Resp5,fLoad


  set Resp1 to []

  put getAt (limCrit,1) into ncb
  put ncb * (ncb-1) / 2 into lim
  repeat with i = 1 to lim
    add Resp1,0
  end repeat

  set Resp2 to []

  put getAt (limCrit,2) into ncb
  put ncb * (ncb-1) / 2 into lim
  repeat with i = 1 to lim
    add Resp2,0
  end repeat

  set Resp3 to []
  put getAt (limCrit,3) into ncb
  put ncb * (ncb-1) / 2 into lim
  repeat with i = 1 to lim
    add Resp3,0
  end repeat

  set Resp4 to []
  put getAt (limCrit,4) into ncb
  put ncb * (ncb-1) / 2 into lim
  repeat with i = 1 to lim
    add Resp4,0
  end repeat

  set Resp5 to []
  put getAt (limCrit,5) into ncb
  put ncb * (ncb-1) / 2 into lim
  repeat with i = 1 to lim
    add Resp5,0
  end repeat

  if fLoad = 0 then carrega
  put fLoad + 1 into fLoad


end inicializa


on montaMatriz i

  global matrizS,nCrit,limCrit,seqSlM,seqRes,seqEla,¬
```

```
        seqTen,seqCus,limCont,Resp1,Resp2,Resp3,Resp4,¬
        Resp5,Mat

  set matrizS to ""

  put (getAt (limCrit,i)) into ordem
  set Mat to abreMatriz (ordem)

  case i of
    1 : set seqV to seqSlM
    2 : set seqV to seqRes
    3 : set seqV to seqEla
    4 : set seqV to seqTen
    5 : set seqV to seqCus
  end case

  case i of
    1 : set RespV to Resp1
    2 : set RespV to Resp2
    3 : set RespV to Resp3
    4 : set RespV to Resp4
    5 : set RespV to Resp5
  end case

  repeat with j = 1 to getAt (limCont,i)
    put getAt (RespV,j) into nor
    put nor * 8 into expand
    if expand > 0 then
      put expand + 1 into expand
    else
      put expand - 1 into expand
    end if
    put expand / abs (expand) into sinal
    put abs (expand) into expand

    put getAt (seqV,(j-1)*2+1) into esq
    put getAt (seqV,(j-1)*2+2) into dir

    if sinal = 1 then
      insere (Mat,ordem,dir,esq,expand)
      insere (Mat,ordem,esq,dir,1/expand)
    else
      insere (Mat,ordem,esq,dir,expand)
      insere (Mat,ordem,dir,esq,1/expand)
    end if
  end repeat

end montaMatriz



on abreMatriz ordem

  put ordem into n

  set Mat to [1]
  repeat with u1 = 1 to n-1
    repeat with u2 = 1 to n
      add Mat,0
```

```
    end repeat
    add Mat,1
  end repeat

  return Mat

end abreMatriz



on insere Mat,ordem,x,y,w

  setAt (Mat,ordem*(x-1)+y,w)

end insere



on le Mat,ordem,x,y

  return getAt (Mat,ordem*(x-1)+y)

end le



on escMat Mat,n,x,y,w

  setAt (Mat,n*(x-1)+y,w)

end escMat



on leMat Mat,n,x,y

  return getAt (Mat,n*(x-1)+y)

end leMat



on norm vet

  put 0 into somat
  repeat with i = 1 to count(vet)
    set somat = somat + getAt (vet,i)
  end repeat

  repeat with i = 1 to count(vet)
    setAt (vet,i,getAt(vet,i)*1.0/somat)
  end repeat

  return vet
```

```
end norm



on media vet

  put 0 into somat
  repeat with i = 1 to count(vet)
    set somat = somat + getAt (vet,i)
  end repeat

  set aux = somat * 1.0 / count (vet)

  return aux

end media



on Saaty Mat

  set n = integer (sqrt(count(Mat)))

  repeat with j = 1 to n
    set vetAux = []
    repeat with i = 1 to n
      add vetAux,leMat (Mat,n,i,j)
    end repeat
    set vetAux = norm (vetAux)
    repeat with i = 1 to n
      escMat (Mat,n,i,j,getAt (vetAux,i))
    end repeat
  end repeat

  set escala = []
  repeat with i = 1 to n
    set vetAux = []
    repeat with j = 1 to n
      add vetAux,leMat (Mat,n,i,j)
    end repeat
    add escala, media (vetAux)
  end repeat

  return norm (escala)

end Saaty



on sorteiaCand

  global candS,opc,idadeMin

  set srts = [[0,0,0,0,0],[0,0,0,0,0],[0,0,0,0,0],¬
             [0,0,0,0,0],[0,0,0,0,0]]
  set qtd = [5,5,5,5,5]
```

```
  set candS = []
  repeat with k = 1 to 10

    set cand = []
    repeat with i = 1 to 5
      set min = 1000
      set max = 0
      set elemMin = 0
      set elemMax = 0
      repeat with j = 1 to getAt (qtd,i)
        if getAt (getAt(srts,i),j) < min then
          set min = getAt (getAt(srts,i),j)
          set elemMin = j
        end if
        if getAt (getAt(srts,i),j) > max then
          set max = getAt (getAt(srts,i),j)
          set elemMax = j
        end if
      end repeat
      set candSort = []
      repeat with j = 1 to getAt (qtd,i)
        if getAt (getAt(srts,i),j) = min then
          add candSort,j
        end if
      end repeat
      set sortNum = random (count(candSort))
      set sort = getAt (candSort,sortNum)
      setAt (getAt(srts,i),sort,getAt(getAt(srts,i),sort) + 1)
      add cand,sort
    end repeat

    add candS,cand

  end repeat

end sorteiaCand
```

Programa de Conversão de Dados de Pesquisa (PCDP)

```
on montaMatriz RespV

  global matrizS,nCrit,limCrit,seqAlt,seqIda,seqEsc,¬
         seqNSE,seqApF,limCont,Resp1,Resp2,Resp3,Resp4,¬
         Resp5,Mat,seqF


  set extr = 4.0
  set ordem = 16
  set seqV = seqF

  set matrizS to ""

  set Mat to abreMatriz (ordem)

  put ordem * (ordem-1) / 2 into limCont

  repeat with j = 1 to 64
    put getAt (RespV,j) into nor

    set nor = nor - extr
    set nor = nor / extr

    put nor * 8 into expand
    if expand > 0 then
      put expand + 1 into expand
    else
      put expand - 1 into expand
    end if
    put expand / abs (expand) into sinal
    put abs (expand) into expand

    put getAt (seqV,(j-1)*2+1) into esq
    put getAt (seqV,(j-1)*2+2) into dir

    if sinal = 1 then
      insere (Mat,ordem,dir,esq,expand)
      insere (Mat,ordem,esq,dir,1/expand)
    else
      insere (Mat,ordem,esq,dir,expand)
      insere (Mat,ordem,dir,esq,1/expand)
    end if
  end repeat

  return Mat

end montaMatriz



on abreMatriz ordem
```

```
   put ordem into n

   set Mat to [1]
   repeat with u1 = 1 to n-1
     repeat with u2 = 1 to n
       add Mat,0
     end repeat
     add Mat,1
   end repeat

   return Mat

end abreMatriz



on insere Mat,ordem,x,y,w

   setAt (Mat,ordem*(x-1)+y,w)

end insere



on le Mat,ordem,x,y

   return getAt (Mat,ordem*(x-1)+y)

end le



on converte RespV

   global Mat,seqF

   set Mt = montaMatriz (RespV)

   set escala = Saaty (Mat)

   return escala

end converte



on automatiza

   global antigo,novo

   set novo = []

   repeat with i = 1 to 24
     add novo, converte (getAt (antigo,i))
   end repeat
```

```
end automatiza



on escMat Mat,n,x,y,w

  setAt (Mat,n*(x-1)+y,w)

end escMat



on leMat Mat,n,x,y

  return getAt (Mat,n*(x-1)+y)

end leMat



on norm vet

  put 0 into somat
  repeat with i = 1 to count(vet)
    set somat = somat + getAt (vet,i)
  end repeat

  repeat with i = 1 to count(vet)
    setAt (vet,i,getAt(vet,i)*1.0/somat)
  end repeat

  return vet

end norm



on media vet

  put 0 into somat
  repeat with i = 1 to count(vet)
    set somat = somat + getAt (vet,i)
  end repeat

  set aux = somat * 1.0 / count (vet)

  return aux

end media
```

Programa Gerador de Configurações de Treinamento

(PGCT)

```
on compila

  global c1,c2,c3,c4,c5,comb,res,i1,i2,i3,i4,i5,T,nExem

  set n = count (comb)

  set nExem = n

  set i1 = []
  set i2 = []
  set i3 = []
  set i4 = []
  set i5 = []
  set  T = []

  repeat with i = 1 to n
    set combin = getAt (comb,i)
    add i1, getAt (c1,getAt (combin,1))
    add i2, getAt (c2,getAt (combin,2))
    add i3, getAt (c3,getAt (combin,3))
    add i4, getAt (c4,getAt (combin,4))
    add i5, getAt (c5,getAt (combin,5))
    add  T, getAt (res,i)
  end repeat

end compila
```

Programa Gerador de Treinamento de Grupo (PGTG)

```
on compila

  global gc1,gc2,gc3,gc4,gc5,gcomb,gres,i1,i2,i3,i4,i5,T,nExem,nPes

  set n = count (getAt (gcomb,1))

  set nExem = n

  set i1 = []
  set i2 = []
  set i3 = []
  set i4 = []
  set i5 = []
  set  T = []

  repeat with k = 1 to nPes
    repeat with i = 1 to nExem
      set combin = getAt (getAt (gcomb,k),i)
      add i1, getAt (getAt (gc1,k),getAt (combin,1))
      add i2, getAt (getAt (gc2,k),getAt (combin,2))
      add i3, getAt (getAt (gc3,k),getAt (combin,3))
      add i4, getAt (getAt (gc4,k),getAt (combin,4))
      add i5, getAt (getAt (gc5,k),getAt (combin,5))
      add  T, getAt (getAt (gres,k),i)
    end repeat
  end repeat

end compila
```

Emulador de RNA (ERNA)

```
on sorteiaSeq

  global seq,nExem

  set usados to []
  repeat with i = 1 to nExem
    add usados,0
  end repeat
  set seq to []
  set cont to 0
  repeat while cont < nExem
    put random (nExem) into aux
    if getAt (usados,aux) = 0 then
      setAt (usados,aux,1)
      add seq,aux
      put cont + 1 into cont
    end if
  end repeat

end sorteiaSeq



on sorteiaW

  global W

  set W to []
  repeat with i = 1 to 22
    put random (200) into aux
    set aux = aux - 100
    set aux = aux / 1000.0
    add W,aux
  end repeat

end sorteiaW



on preparaIteracao

  global ite,ex,EQM,acEsc

  set ex  = 0
  set EQM = 0
  put ite + 1 into ite

  sorteiaSeq

  if acEsc = 0 then
```

```
    mostraIteracao
  end if

end preparaIteracao



on preparaBP

  global datGraf

  normaliza
  sorteiaW
  mostraTaxTolMom
  mostraW

  set datGraf = []

end preparaBP



on BP

  global W,taxa,tol,mom,ite,ex,EQM,nEntr,nSaid,nExem,¬
         i1,i2,i3,i4,i5,T,o,dW1,dW2,seq,davez,acLuz,¬
         h,d,e,EQMOld,meiaVida,datGraf,acEsc,arr

  if ex < nExem then

    set ex = ex + 1
    set davez = getAt (seq,ex)

    if acEsc = 0 then
      mostraExemplo
      mostraEntradas
    end if


    if acLuz = 1 then forward1

    set h = []
    repeat with i = 1 to 3

      set somat = 0

      set par = getAt (i1,davez) * getAt (W,0+i)
      set somat = somat + par

      set par = getAt (i2,davez) * getAt (W,3+i)
      set somat = somat + par

      set par = getAt (i3,davez) * getAt (W,6+i)
      set somat = somat + par

      set par = getAt (i4,davez) * getAt (W,9+i)
      set somat = somat + par
```

```
    set par = getAt (i5,davez) * getAt (W,12+i)
    set somat = somat + par

    set par = 1.0 * getAt (W,15+i)
    set somat = somat + par

    add h, funcAtiv (somat)

  end repeat


  add h, 1.0


  if acLuz = 1 then forward2

  set somat = 0
  repeat with i = 1 to 4
    set par = getAt (h,i) * getAt (W,18+i)
    set somat = somat + par
  end repeat


  set o = funcAtiv (somat)


  if acLuz = 1 then forward3

  set aux = getAt (T,davez) - o
  set dq = power (aux,2)

  set EQM = EQM * 1.0 * (ex-1) / ex + dq * 1.0 / ex

  if acEsc = 0 then
    mostraSaida
    mostraEQM
  end if


  if acLuz = 1 then backward1


  set d = o * (1 - o) * (getAt (T,davez) - o)

  set e = []
  repeat with i = 1 to 4
    set aux = getAt (h,i) * (1 - getAt (h,i))
    set aux = aux * getAt (W,18+i)
    set aux = aux * d
    add e, aux
  end repeat

  repeat with i = 1 to 4
    set aux = taxa * getAt (h,i) * d
    set aux = aux + mom * getAt (dW2,i)
    setAt (dW2,i,aux)
    setAt (W, 18+i, getAt(W,18+i) + getAt (dW2,i))
  end repeat
```

```
  if acLuz = 1 then backward2

  repeat with i = 1 to 3

    set aux = taxa * getAt (i1,davez) * getAt (e,i)
    set aux = aux + mom * getAt (dW1,0+i)
    setAt (dW1,0+i,aux)
    setAt (W, 0+i, getAt(W,0+i) + getAt (dW1,0+i))

    set aux = taxa * getAt (i2,davez) * getAt (e,i)
    set aux = aux + mom * getAt (dW1,3+i)
    setAt (dW1,3+i,aux)
    setAt (W, 3+i, getAt(W,3+i) + getAt (dW1,3+i))

    set aux = taxa * getAt (i3,davez) * getAt (e,i)
    set aux = aux + mom * getAt (dW1,6+i)
    setAt (dW1,6+i,aux)
    setAt (W, 6+i, getAt(W,6+i) + getAt (dW1,6+i))

    set aux = taxa * getAt (i4,davez) * getAt (e,i)
    set aux = aux + mom * getAt (dW1,9+i)
    setAt (dW1,9+i,aux)
    setAt (W, 9+i, getAt(W,9+i) + getAt (dW1,9+i))

    set aux = taxa * getAt (i5,davez) * getAt (e,i)
    set aux = aux + mom * getAt (dW1,12+i)
    setAt (dW1,12+i,aux)
    setAt (W, 12+i, getAt(W,12+i) + getAt (dW1,12+i))

    set aux = taxa * 1.0 * getAt (e,i)
    set aux = aux + mom * getAt (dW1,15+i)
    setAt (dW1,15+i,aux)
    setAt (W, 15+i, getAt(W,15+i) + getAt (dW1,15+i))

  end repeat


  if acEsc = 0 then
    mostraW
  end if


  if acLuz = 1 then backward3

else


  if ite = 1 then add datGraf,EQM


  ---------------------------

  if ite mod meiaVida = 0 then
    if (EQMOld - EQM) / EQMOld < 0.001 then
      set taxa = taxa * arr
      set mom  = mom  * arr
      mostraTaxTolMom
    end if
```

```
      set EQMOld = EQM

      add datGraf,EQM

      if acEsc = 1 then
        mostraIteracao
        mostraExemplo
        mostraW
        mostraEQM
      end if

      if EQM < tol then atingiu

    end if

    ----------------------------


    set ex = 0

  end if


end BP



on funcAtiv par

  set f = 1.0 / (1.0 + exp (-1.0 * par))

  return f

end funcAtiv



on normaliza

  global i1,i2,i3,i4,i5,T,nExem,z,vetMinMax

  set norMin = 0.20
  set norMax = 0.80


  repeat with i = 1 to nExem
    setAt (i1,i, power (getAt (i1,i),z))
    setAt (i2,i, power (getAt (i2,i),z))
    setAt (i3,i, power (getAt (i3,i),z))
    setAt (i4,i, power (getAt (i4,i),z))
    setAt (i5,i, power (getAt (i5,i),z))
    setAt (T,i, power (getAt (T,i),z))
  end repeat


  set vetMinMax = []
```

```
set min = 1
set max = 0
repeat with i = 1 to nExem
  if getAt (i1,i) < min then set min = getAt (i1,i)
  if getAt (i1,i) > max then set max = getAt (i1,i)
end repeat
if max <> min then
  repeat with i = 1 to nExem
    setAt (i1,i,(getAt(i1,i) - min) * 1.0 / (max - min) ¬
          * (norMax - norMin) + norMin)
  end repeat
else
  repeat with i = 1 to nExem
    setAt (i1,i,0)
  end repeat
end if

add vetMinMax,min
add vetMinMax,max


set min = 1
set max = 0
repeat with i = 1 to nExem
  if getAt (i2,i) < min then set min = getAt (i2,i)
  if getAt (i2,i) > max then set max = getAt (i2,i)
end repeat
if max <> min then
  repeat with i = 1 to nExem
    setAt (i2,i,(getAt(i2,i) - min) * 1.0 / (max - min) ¬
          * (norMax - norMin) + norMin)
  end repeat
else
  repeat with i = 1 to nExem
    setAt (i2,i,0)
  end repeat
end if

add vetMinMax,min
add vetMinMax,max


set min = 1
set max = 0
repeat with i = 1 to nExem
  if getAt (i3,i) < min then set min = getAt (i3,i)
  if getAt (i3,i) > max then set max = getAt (i3,i)
end repeat
if max <> min then
  repeat with i = 1 to nExem
    setAt (i3,i,(getAt(i3,i) - min) * 1.0 / (max - min) ¬
          * (norMax - norMin) + norMin)
  end repeat
else
  repeat with i = 1 to nExem
    setAt (i3,i,0)
  end repeat
end if

add vetMinMax,min
```

```
add vetMinMax,max


set min = 1
set max = 0
repeat with i = 1 to nExem
  if getAt (i4,i) < min then set min = getAt (i4,i)
  if getAt (i4,i) > max then set max = getAt (i4,i)
end repeat
if max <> min then
  repeat with i = 1 to nExem
    setAt (i4,i,(getAt(i4,i) - min) * 1.0 / (max - min) ¬
          * (norMax - norMin) + norMin)
  end repeat
else
  repeat with i = 1 to nExem
    setAt (i4,i,0)
  end repeat
end if

add vetMinMax,min
add vetMinMax,max


set min = 1
set max = 0
repeat with i = 1 to nExem
  if getAt (i5,i) < min then set min = getAt (i5,i)
  if getAt (i5,i) > max then set max = getAt (i5,i)
end repeat
if max <> min then
  repeat with i = 1 to nExem
    setAt (i5,i,(getAt(i5,i) - min) * 1.0 / (max - min) ¬
          * (norMax - norMin) + norMin)
  end repeat
else
  repeat with i = 1 to nExem
    setAt (i5,i,0)
  end repeat
end if

add vetMinMax,min
add vetMinMax,max


set min = 1
set max = 0
repeat with i = 1 to nExem
  if getAt (T,i) < min then set min = getAt (T,i)
  if getAt (T,i) > max then set max = getAt (T,i)
end repeat
if max <> min then
  repeat with i = 1 to nExem
    setAt (T,i,(getAt(T,i) - min) * 1.0 / (max - min) ¬
          * (norMax - norMin) + norMin)
  end repeat
else
  repeat with i = 1 to nExem
    setAt (T,i,0)
  end repeat
```

```
  end if

  add vetMinMax,min
  add vetMinMax,max


end normaliza



on preparaTeste

  global espeGraf,saidGraf,ex

  set ex = 0
  normalizaTeste

  mostraExemplo

  set espeGraf = []
  set saidGraf = []

end preparaTeste



on Teste

  global W,taxa,tol,mom,ite,ex,EQM,nEntr,nSaid,nExem,¬
         i1,i2,i3,i4,i5,T,o,dW1,dW2,seq,davez,acLuz,¬
         h,d,e,EQMOld,acEsc,espeGraf,saidGraf

  if ex < nExem then

    set ex = ex + 1
    set davez = ex

    mostraExemplo
    mostraEntradas


    if acLuz = 1 then forward1

    set h = []
    repeat with i = 1 to 3

      set somat = 0

      set par = getAt (i1,davez) * getAt (W,0+i)
      set somat = somat + par

      set par = getAt (i2,davez) * getAt (W,3+i)
      set somat = somat + par

      set par = getAt (i3,davez) * getAt (W,6+i)
      set somat = somat + par

      set par = getAt (i4,davez) * getAt (W,9+i)
```

```
      set somat = somat + par

      set par = getAt (i5,davez) * getAt (W,12+i)
      set somat = somat + par

      set par = 1.0 * getAt (W,15+i)
      set somat = somat + par

      add h, funcAtiv (somat)

    end repeat


    add h, 1.0


    if acLuz = 1 then forward2

    set somat = 0
    repeat with i = 1 to 4
      set par = getAt (h,i) * getAt (W,18+i)
      set somat = somat + par
    end repeat


    set o = funcAtiv (somat)


    if acLuz = 1 then forward3

    set aux = getAt (T,davez) - o
    set dq = power (aux,2)

    set EQM = EQM * 1.0 * (ex-1) / ex + dq * 1.0 / ex

    mostraSaida
    mostraEQM

    add espeGraf, getAt (T,davez)
    add saidGraf, o


  else


    set ex = 0

    salvaTeste

  end if

end Teste



on normalizaTeste
```

```
global i1,i2,i3,i4,i5,T,nExem,z,vetMinMax

set norMin = 0.20
set norMax = 0.80


repeat with i = 1 to nExem
  setAt (i1,i, power (getAt (i1,i),z))
  setAt (i2,i, power (getAt (i2,i),z))
  setAt (i3,i, power (getAt (i3,i),z))
  setAt (i4,i, power (getAt (i4,i),z))
  setAt (i5,i, power (getAt (i5,i),z))
  setAt (T,i, power (getAt (T,i),z))
end repeat


set min = getAt (vetMinMax,1)
set max = getAt (vetMinMax,2)
if max <> min then
  repeat with i = 1 to nExem
    setAt (i1,i,(getAt(i1,i) - min) * 1.0 / (max - min) ¬
           * (norMax - norMin) + norMin)
  end repeat
else
  repeat with i = 1 to nExem
    setAt (i1,i,0)
  end repeat
end if


set min = getAt (vetMinMax,3)
set max = getAt (vetMinMax,4)
if max <> min then
  repeat with i = 1 to nExem
    setAt (i2,i,(getAt(i2,i) - min) * 1.0 / (max - min) ¬
           * (norMax - norMin) + norMin)
  end repeat
else
  repeat with i = 1 to nExem
    setAt (i2,i,0)
  end repeat
end if


set min = getAt (vetMinMax,5)
set max = getAt (vetMinMax,6)
if max <> min then
  repeat with i = 1 to nExem
    setAt (i3,i,(getAt(i3,i) - min) * 1.0 / (max - min) ¬
           * (norMax - norMin) + norMin)
  end repeat
else
  repeat with i = 1 to nExem
    setAt (i3,i,0)
  end repeat
end if


set min = getAt (vetMinMax,7)
set max = getAt (vetMinMax,8)
```

```
  if max <> min then
    repeat with i = 1 to nExem
      setAt (i4,i,(getAt(i4,i) - min) * 1.0 / (max - min) ¬
             * (norMax - norMin) + norMin)
    end repeat
  else
    repeat with i = 1 to nExem
      setAt (i4,i,0)
    end repeat
  end if


  set min = getAt (vetMinMax,9)
  set max = getAt (vetMinMax,10)
  if max <> min then
    repeat with i = 1 to nExem
      setAt (i5,i,(getAt(i5,i) - min) * 1.0 / (max - min) ¬
             * (norMax - norMin) + norMin)
    end repeat
  else
    repeat with i = 1 to nExem
      setAt (i5,i,0)
    end repeat
  end if


  set min = getAt (vetMinMax,11)
  set max = getAt (vetMinMax,12)
  if max <> min then
    repeat with i = 1 to nExem
      setAt (T,i,(getAt(T,i) - min) * 1.0 / (max - min) ¬
             * (norMax - norMin) + norMin)
    end repeat
  else
    repeat with i = 1 to nExem
      setAt (T,i,0)
    end repeat
  end if


end normalizaTeste



on preparaCalculo

  global saidGraf,ex

  set ex = 0

  normalizaCalculo

  mostraExemplo

  set saidGraf = []

end preparaCalculo
```

```
on calculo

  global W,taxa,tol,mom,ite,ex,EQM,nEntr,nSaid,nExem,¬
         i1,i2,i3,i4,i5,T,o,dW1,dW2,seq,davez,acLuz,¬
         h,d,e,EQMOld,acEsc,saidGraf

  if ex < nExem then

    set ex = ex + 1
    set davez = ex

    if acEsc = 0 then
      mostraExemplo
      mostraEntradas
    end if


    if acLuz = 1 then forward1

    set h = []
    repeat with i = 1 to 3

      set somat = 0

      set par = getAt (i1,davez) * getAt (W,0+i)
      set somat = somat + par

      set par = getAt (i2,davez) * getAt (W,3+i)
      set somat = somat + par

      set par = getAt (i3,davez) * getAt (W,6+i)
      set somat = somat + par

      set par = getAt (i4,davez) * getAt (W,9+i)
      set somat = somat + par

      set par = getAt (i5,davez) * getAt (W,12+i)
      set somat = somat + par

      set par = 1.0 * getAt (W,15+i)
      set somat = somat + par

      add h, funcAtiv (somat)

    end repeat


    add h, 1.0


    if acLuz = 1 then forward2

    set somat = 0
    repeat with i = 1 to 4
      set par = getAt (h,i) * getAt (W,18+i)
      set somat = somat + par
    end repeat
```

```
    set o = funcAtiv (somat)


    if acLuz = 1 then forward3

    if acEsc = 0 then
      mostraSoSaida
    end if

    add saidGraf, o

  else
    set ex = 0
    salvaCalculo
  end if


end calculo



on normalizaCalculo

  global i1,i2,i3,i4,i5,nExem,z,vetMinMax

  set norMin = 0.20
  set norMax = 0.80


  repeat with i = 1 to nExem
    setAt (i1,i, power (getAt (i1,i),z))
    setAt (i2,i, power (getAt (i2,i),z))
    setAt (i3,i, power (getAt (i3,i),z))
    setAt (i4,i, power (getAt (i4,i),z))
    setAt (i5,i, power (getAt (i5,i),z))
  end repeat

  set min = getAt (vetMinMax,1)
  set max = getAt (vetMinMax,2)
  if max <> min then
    repeat with i = 1 to nExem
      setAt (i1,i,(getAt(i1,i) - min) * 1.0 / (max - min) ¬
            * (norMax - norMin) + norMin)
    end repeat
  else
    repeat with i = 1 to nExem
      setAt (i1,i,0)
    end repeat
  end if


  set min = getAt (vetMinMax,3)
  set max = getAt (vetMinMax,4)
  if max <> min then
    repeat with i = 1 to nExem
      setAt (i2,i,(getAt(i2,i) - min) * 1.0 / (max - min) ¬
            * (norMax - norMin) + norMin)
    end repeat
```

```
  else
    repeat with i = 1 to nExem
      setAt (i2,i,0)
    end repeat
  end if


  set min = getAt (vetMinMax,5)
  set max = getAt (vetMinMax,6)
  if max <> min then
    repeat with i = 1 to nExem
      setAt (i3,i,(getAt(i3,i) - min) * 1.0 / (max - min) ¬
             * (norMax - norMin) + norMin)
    end repeat
  else
    repeat with i = 1 to nExem
      setAt (i3,i,0)
    end repeat
  end if


  set min = getAt (vetMinMax,7)
  set max = getAt (vetMinMax,8)
  if max <> min then
    repeat with i = 1 to nExem
      setAt (i4,i,(getAt(i4,i) - min) * 1.0 / (max - min) ¬
             * (norMax - norMin) + norMin)
    end repeat
  else
    repeat with i = 1 to nExem
      setAt (i4,i,0)
    end repeat
  end if


  set min = getAt (vetMinMax,9)
  set max = getAt (vetMinMax,10)
  if max <> min then
    repeat with i = 1 to nExem
      setAt (i5,i,(getAt(i5,i) - min) * 1.0 / (max - min) ¬
             * (norMax - norMin) + norMin)
    end repeat
  else
    repeat with i = 1 to nExem
      setAt (i5,i,0)
    end repeat
  end if


end normalizaCalculo
```

REFERÊNCIAS BIBLIOGRÁFICAS

ADELI, H.; HUNG, S. "Machine Learning - Neural Networks, Genetic Algorithms, and Fuzzy Systems". John Wiley & Sons. New York, 1995.

ADELMAN, L. "Evaluating Decision Support and Expert Systems". John Wiley & Sons. New York, 1992.

BLUM, A. "Neural Networks in C++ : An Object-oriented Framework for building Connectionist Systems". John Wiley & Sons. New York, 1992.

CASEY, J. "Picking the Right Expert System Application". AI Expert, 4 (9): 44-47, 1989.

FRIEDHOFER, M. "Implementation of a Fuzzy-based Decision Support System : A do-it-yourself recipe".

KARTALOPOULOS, S. V. "Understanding Neural Networks and Fuzzy Logic - Basic Concepts and Applications". IEEE Press. New York, 1996.

KOVÁCS, Z. L. "Redes Neurais Artificiais - Fundamentos e Aplicações". 1996

MADUREIRA, O. M. "Planejamento e Desenvolvimento de Produtos". Apostila do Curso. 1995.

SAATY, T. L. "A Scaling Method for Priorities in Hierarchical Structures". Journal of Mathematical Psychology 15, 234-281, 1977.

SIMÕES, M. G.; FRIEDHOFER, M. "An improved Fuzzy Based Algorithm for Industrial Decision Support"

SIMÕES, M. G.; FRIEDHOFER, M. "Decision Criteria with Different Degrees of Importance : a Contrast Diffusion/Intensification Approach"

WOLFGRAM, D. D.; DEAR, T. J.; GALBRAITH, C. S. "Expert Systems for the Technical Professional". New York: Wiley, 1987.