CENTRO UNIVERSITÁRIO DA FEI

Julio André Sgarbi

DOMÓTICA INTELIGENTE: AUTOMAÇÃO RESIDENCIAL BASEADA EM COMPORTAMENTO

São Bernardo do Campo

2007

Julio André Sgarbi

DOMÓTICA INTELIGENTE: AUTOMAÇÃO RESIDENCIAL BASEADA EM COMPORTAMENTO

Dissertação de Mestrado apresentada ao Centro Universitário da FEI como parte dos requisitos necessários para a obtenção do título de Mestre em Engenharia Elétrica.
Orientador: Prof. Dr. Flavio Tonidandel

São Bernardo do Campo

2007

Sgarbi, Julio André

Domótica Inteligente: Automação Residencial Baseada em Comportamento / Julio André Sgarbi – São Bernardo do Campo, 2007.

107 f.: il.

Dissertação de Mestrado – Centro Universitário da FEI.

Orientador: Prof. Dr. Flavio Tonidandel.

1. Automação Residencial – Aprendizado. 2. Domótica Inteligente. 3. Inteligência artificial. I.Tonidandel, Flavio, orient. II.Título.

CDU 681.3.06

A Deus, meus pais, minha amada esposa e meu amado filho.

AGRADECIMENTOS

Agradeço a todos os que me deram apoio e ensinaram a ter perseverança durante este período de trabalho desta dissertação.

Agradeço a Deus por ter iluminado meu caminho me dando saúde, perseverança e a certeza de que é possível realizar nossos sonhos.

Aos meus pais que sempre me ajudaram nas dificuldades da vida.

A minha esposa Paula Valéria que me incentivou, nos momentos difíceis, não me deixando desistir nunca. Agradeço pela maior alegria da minha vida, ocorrida durante a elaboração deste trabalho, o nosso filho Rafael.

Aos amigos do mestrado, sem exceção, que sempre foram unidos nos momentos difíceis das aulas e nos momentos de alegria e descontração.

Aos professores da FEI, que sempre foram amigos e me proporcionaram excelentes momentos de aprendizado.

Ao meu orientador professor Dr. Flavio Tonidandel, por ter aceitado me orientar neste trabalho, com paciência, disposição, sempre me conduzindo pelo caminho da reflexão. Agradeço pelas valiosas contribuições para o meu desenvolvimento intelectual e pessoal.

Aos professores Dr. Márcio Rillo e Dr. José Reinaldo Silva, que participaram da minha banca de qualificação, pelas valiosas contribuições para o desenvolvimento desta dissertação.

A todos aqueles que direta ou indiretamente colaboraram para que o desenvolvimento deste trabalho fosse possível.

Algo só é impossível até que alguém duvide e acabe provando o contrário.

Albert Einstein

RESUMO

A automação residencial evoluiu muito nos últimos anos, entretanto pouco destaque é dado à automação residencial baseada no comportamento dos habitantes. O sistema proposto ABC+ (Automação Baseada em Comportamento) foi desenvolvido para observar e aprender regras em uma casa de acordo com o comportamento de seu habitante, utilizando o conceito de aprendizado com regras de indução.

O principal problema abordado neste trabalho foi desenvolver um sistema simples e amigável que abrangesse as várias particularidades envolvidas na automação inteligente de uma residência, tais como as seqüências causais de eventos, que geram regras indesejáveis; a inserção de novas regras para os habitantes, sem causar desconforto aos mesmos; os diferentes perfis de habitantes e ambientes, entre outros.

As experiências foram feitas através de dois simuladores desenvolvidos para se comprovar primeiramente o correto funcionamento do sistema proposto e posteriormente observar o comportamento do sistema e suas variáveis quando submetidos à ação de um agente (habitante).

Palavras-chave: Inteligência Artificial, Automação Residencial, Domótica Inteligente, Aprendizado.

ABSTRACT

The residential automation evolved in the last years, however little importance is given to residential automation based in the behavior of the inhabitants. The system ABC+ (Automation based on behavior) was developed to observe and to learn rules in a house according to the behavior of its inhabitant, using the concept of learning with induction rules.

The main problem approached in this work was to develop a simple and friendly system that enclosed the some particularities involved in intelligent automation of a residence, such as the causal sequences of events, which generate undesirable rules; the insertion of new rules for the inhabitants, without causing discomfort to the inhabitants; the different profiles of inhabitants and environments, among others.

The experiences had been made through two simulators, developed to prove the correct functioning of the system and to observe the behavior of the system and its variables when submitted to the action of an agent (inhabitant).

Keywords: Artificial Intelligence, Residential Automation, Intelligent Domotics, Learning.

SUMÁRIO

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE SÍMBOLOS

Δt – Intervalo de tempo.

Σ – Somatória.

LISTA ABREVIATURAS

ABC – Automação Baseada em Comportamento.

AC - Número de Regras Ativas criadas.

AE - Número de Regras Ativas eliminadas (rebaixadas).

AI - Artificial Intelligent.

ATIV – Campo do BDAtivas utilizado para pontuar positivamente uma regra.

ATIV_At - Valor do campo ATIV do BDAtivas para regra ir para o BDEmbrio, devido a desuso.

BD – Bando de Dados.

BDAtivas – Banco de Dados de Regras Ativas.

BDEmbrio – Banco de Dados de Regras Embrionárias.

BDEventos – Banco de Dados de Eventos por Atuador.

CO2 – Dióxido de Carbono.

EC - Número de Regras Embrionárias criadas.

EE - Número de Regras Embrionárias eliminadas.

EXC - Campo do BDAtivas utilizado para pontuar negativamente uma regra.

EXC_At - Valor do campo EXC do BDAtivas para excluir a regra.

GPS - Global Positioning System.

IA – Inteligência Artificial.

LDC - Linguagem de Descrição de Conhecimento de Fundo (domínio).

LDE - Linguagem de Descrição de Exemplos.

LDH - Linguagem de Descrição de Hipóteses (Conceitos).

MIT - Massachusetts Institute of Technology.

NOK - Campo do BDEmbrio utilizado para excluir uma regra.

NOK_Emb - Valor do campo NOK do BDEmbrio para regra ser excluída.

NuEv - Número de eventos no BDEventos para acionar o C4.5.

OK - Campo do BDEmbrio utilizado para validar uma regra.

OK_Emb - Valor do campo OK do BDEmbrio para regra virar Regra Ativa.

OK_Emb_exc - Valor do campo OK do BDEmbrio para a regra ser excluída por desuso.

PIR – Passive Infra Red.

RBC - Raciocínio Baseado em Casos.

RFID – Radio Frequency Identification.

TDIDT - Top-Down Induction Trees.

1 INTRODUÇÃO

Existe uma série de trabalhos que abordam o tema automação residencial. Os avanços tecnológicos e a busca por conforto e segurança vêm fazendo com que a cada dia as residências tenham mais e melhores sistemas automatizadores.

Algumas vezes estes sistemas são implementados com técnicas de inteligência artificial, entretanto a maioria deles possui apenas mecanismos automáticos (BOLZANI, 2004a).

A principal ênfase destes sistemas está no gerenciamento de recursos, segurança, conforto e atualmente em entretenimento (MURATORI, 2005).

Muitos dos trabalhos publicados alegam que seus sistemas são inteligentes. Alguns deles detectam eventos como a presença dos habitantes e suas ações, ou possuem sensores que conseguem capturar mudanças nas condições do ambiente; a estes eventos são aplicadas reações que estão previamente configuradas (BOLZANI, 2004a). O conceito de inteligência deve ir além de automatizar ou simplesmente aplicar regras pré-estabelecidas; os sistemas inteligentes devem interagir com os habitantes da casa e aprender com seus comportamentos.

Automação residencial deve evoluir para o conceito de Domótica Inteligente, no qual se entende que os dados obtidos pelos sensores da casa devem ser analisados de modo a adaptar as regras de automação do ambiente ao comportamento dos habitantes (TONIDANDEL; TAKIUCHI; MELO, 2004).

Os seres humanos estão em constante mudança; o que é uma regra ou rotina hoje, amanhã pode não ser. Os hábitos, horários e atividades mudam com o passar do tempo. Os sistemas têm de aprender e se adaptar a isto.

Neste trabalho será apresentado um sistema de automação residencial inteligente (Sistema ABC+, Automação Baseada em Comportamento) que cria regras para ação de atuadores em função do aprendizado com o comportamento dos habitantes de uma casa, dando ênfase à obtenção, tratamento e manutenção das regras.

1.1 Objetivo

O objetivo principal deste trabalho é o desenvolvimento de um sistema de automação residencial baseado no comportamento do habitante, com ênfase no funcionamento do sistema e na análise das variáveis do sistema quando este é submetido à ação de um agente (habitante).

1.2 Motivação

A motivação pela escolha do tema ocorre ao se observar a dificuldade que algumas pessoas tem para programar ou configurar certos aparelhos e sistemas. Se para estas pessoas programar um simples vídeo cassete é quase impossível, imagine um sistema de automação residencial. Assim, torna-se necessário a criação de um sistema de automação residencial que aprenda o comportamento do habitante e que seja simples. Em termos práticos o sistema deve permitir que pessoas com menor familiarização aos sistemas domóticos possam utilizá-los, devido à diminuição na programação dos mesmos.

Um sistema pode guardar eventos e imediatamente transformar estes eventos em regras, entretanto a quantidade de regras aumentará a cada evento que surgir, ademais, soma-se a isto a quantidade de memória e processamento necessários para o sistema funcionar, o mesmo torna-se impraticável. Por este motivo, torna-se necessário utilizar alguma técnica que permita reduzir a quantidade de regras, ou melhor, avaliar os eventos e reuní-los em poucas regras. A técnica utilizada para isto neste trabalho será a da indução de árvores de decisão.

O uso de um simulador visa conhecer o funcionamento do sistema proposto, permitindo corrigir possíveis falhas e entender em detalhe o comportamento dos parâmetros do mesmo.

1.3 Organização do Trabalho

Este trabalho está composto da seguinte forma:

- O capítulo 2 trata o tema Aprendizado de Máquina, dando ênfase aos sistemas de aprendizado automático e ao algoritmo C4.5.
- No capítulo 3 é feita a descrição de domótica e do sistema ABC.
- O capítulo 4 detalha o sistema ABC+ proposto.
- No capítulo 5 são descritas as simulações realizadas no trabalho, os resultados obtidos e suas devidas análises.
- O capítulo 6 encerra o trabalho com a conclusão e possíveis trabalhos futuros.

2 APRENDIZADO DE MÁQUINA

Este capítulo apresenta os principais tópicos de Aprendizado de Máquina, nos quais está fundamentado o trabalho, é nele que está desenvolvida a revisão da literatura sobre o tema, mostrando os conceitos, métodos e outros trabalhos utilizados como base para este trabalho.

2.1 Sistemas de Aprendizado Automático

Os Sistemas de Aprendizado Automático são um tema amplo, abordado dentro da área de Inteligência Artificial.

2.1.1 Inteligência Artificial

A expressão "*Artificial Intelligent*", também conhecida como AI (IA em português), foi estabelecida em 1956 por McCarthy, e posteriormente abordada por pesquisadores como Minsky, Simon, Newell e outros.

Existem muitas definições de Inteligência Artificial, não se encontra uma definição única e padrão; pode-se dizer que Inteligência Artificial é uma área de pesquisa da Ciência da Computação voltada ao estudo de métodos ou dispositivos computacionais que tenham ou simulem a capacidade humana de resolução de problemas, de pensar ou ser inteligente.

2.1.2 Aprendizado

O aprendizado também é um conceito difícil de ser definido. Aprendizado é a capacidade de se adaptar, de modificar e melhorar seu comportamento e suas respostas, o que é uma das propriedades mais importantes dos seres ditos inteligentes (OSÓRIO, 1999).

A capacidade de aprender está ligada diretamente a alguns itens: adaptação e mudança de comportamento de maneira a evoluir; correção dos erros cometidos no passado, de modo a não repeti-los no futuro; melhoria do desempenho do sistema como um todo; interação com o meio, pois é através deste contato com o mundo que nos cerca que se podem trocar experiências e assim adquirir novos conhecimentos; representação do conhecimento adquirido - guardar uma massa muito grande de conhecimentos requer uma forma de representar estes conhecimentos que permita ao sistema explorá-los de maneira conveniente (OSÓRIO, 1999).

O aprendizado é o motivo que levou à superioridade da inteligência humana (SANCHES; GEROMINI, 2001), é a essência da inteligência. Em face disto deve-se aumentar a capacidade de aprendizado das máquinas caso se queira que elas tenham comportamento inteligente.

O Homem nasce apto para iniciar seu aprendizado, ampliando seu conhecimento através de reordenações sucessivas, entretanto as máquinas não possuem programa inicial para aprender, precisam de um para fazê-lo.

2.1.3 Aprendizado Automático

O aprendizado automático (*Machine Learning*) ou de máquina é uma área de IA que busca métodos computacionais ligados a aquisição de novo conhecimento e formas de organizar o conhecimento já existente.

Segundo Simon (1983):

> *"Aprendizado denota mudanças no sistema, que são adaptáveis no sentido de que elas possibilitam que o sistema faça a mesma tarefa ou tarefas sobre uma mesma população, de uma maneira mais eficiente a cada vez."*

Esta definição abrange de um lado o refinamento de habilidades, no qual se entende que a prática sucessiva de tarefas pode tornar melhor a execução das mesmas, e do outro lado está a aquisição de conhecimento, o qual é adquirido geralmente através da experiência.

Os sistemas de aprendizado automático devem ter a habilidade de não somente acumular conhecimento, mas também de absorvê-lo ou generalizá-lo.

Visto de uma maneira prática, o aprendizado automático pode ser entendido como um conjunto de métodos, técnicas e ferramentas próprias para a aquisição automatizada de conhecimento a partir de conjuntos de dados (MITCHELL, 1997).

Enfim, o aprendizado automático é o campo dedicado ao desenvolvimento de métodos computacionais para os processos de aprendizagem e a aplicação dos sistemas informáticos de aprendizagem a problemas práticos (MICHALSKY; BRATKO; KUBAT, 1998).

Segundo Mitchell (1997), o aprendizado automático enfrenta o desafio de construir sistemas computacionais que automaticamente melhorem com a experiência, sendo essencial para isto definir o problema de aprendizado de maneira certa. Ele apresenta a seguinte definição, onde constam três partes fundamentais:

> *"Pode-se afirmar que um programa computacional é capaz de aprender a partir da experiência E com respeito a um grupo de tarefas T e segundo a medida de desempenho P, se seu desempenho nas tarefas T, medido segundo P, melhora com a experiência E."* (MITCHELL, 1997).

É desejável que os sistemas respondam melhor aos seus objetivos com o aprendizado, entretanto isto não está garantido. O aprendizado possibilita levar a melhora.

A seguir serão apresentadas as características que possibilitam classificar os sistemas de Aprendizado Automático quanto ao paradigma, modo, forma de aprendizado e linguagens de descrição.

2.1.4 Paradigmas de Aprendizado Automático

Existem vários paradigmas de Aprendizado Automático (BATISTA, 2003): baseado em exemplos, estatísticos, conexionista, genético (ou evolutivo) e simbólico.

Os sistemas baseados em exemplos ou *lazy* caracterizam-se por classificar um exemplo lembrando de outro similar cuja classe é conhecida e assumem que este novo exemplo terá a mesma classe, assim classificam-se casos nunca vistos através de casos similares conhecidos (AHA; KIBLER; ALBERT, 1991).

Suas características principais são: Identificar e reter casos protótipos que juntos resumam toda a informação importante; a similaridade entre os casos pode ser medida, se os atributos forem contínuos, calculando a distância entre dois casos como a raiz quadrada da soma dos quadrados da diferença dos atributos; Um novo caso é relacionado com o mais próximo dos casos armazenados ou levando-se em consideração os diferentes graus de similaridade entre cada caso. As técnicas mais conhecidas desse paradigma são, provavelmente, *Nearest Neighbours* e Raciocínio Baseado em Casos (RBC), como exemplo deste último pode-se citar o sistema FAR-OFF (TONIDANDEL, 2003).

Já os sistemas estatísticos consistem em utilizar modelos estatísticos para encontrar uma boa aproximação do conceito induzido. Muitos destes modelos são paramétricos e assumem alguma forma de modelo, encontrando valores apropriados para os parâmetros do modelo a partir de dados.

Como exemplo, um classificador linear assume que classes podem ser expressas como combinação linear dos valores dos atributos e então se procura uma combinação linear particular que fornece a melhor aproximação sobre o conjunto de dados a serem classificados.

Outros exemplos são o algoritmo CART (*Classification and Regression Tree*) (BREIMAN; FREDMAN; OLSHEN, 1984), o qual é um sistema desenvolvido por estatísticos para montar árvores de decisão, e o QUEST (*Quick, Unbiased, Efficient, Statistical Tree*) (LOH e SHIH, 1997), um algoritmo estatístico que seleciona variáveis sem vício e constrói árvores binárias precisas de maneira rápida e eficaz.

As redes neurais são exemplos de sistemas conexionistas. Elas são construções matemáticas simplificadas inspiradas no modelo biológico do sistema nervoso. Sua representação envolve unidades altamente interconectadas, no qual o nome conexionismo é utilizado para descrever a área de estudo.

McCulloch e Pitts (1943) foram os pioneiros a iniciar pesquisas com redes neurais. McCulloch foi um psiquiatra que pesquisou por 20 anos uma forma de representar um evento no sistema nervoso; Pitts era um jovem pesquisador que se juntou a McCulloch em 1942. Cerca de quinze anos após a publicação de McCulloch e Pitts, Rosenblatt (1958) apresentou o perceptron. Após isto, Minsky e Papert em seu livro *Perceptrons* (MINSKY; PAPERT, 1969) demonstraram a existência de limites nos perceptrons de uma camada, fato que diminuiu o interesse nos estudos sobre redes neurais. Hopfield (1982) apresentou uma nova forma de compreender os cálculos realizados em redes recorrentes com conexões sinápticas simétricas, utilizando para isto a idéia de uma função de energia, fazendo assim retornar o desenvolvimento do estudo das redes neurais.

Os fatos mais recentes em redes neurais são o desenvolvimento do algoritmo *backpropagation* que supera os problemas dos Perceptrons, em 1986 por Rumelhart, Hinton e Williams e a descrição do procedimento para projeto de redes utilizando funções de base radial, em 1988 por Broomhead e Lowe.

Além dos paradigmas citados acima, tem-se ainda o paradigma genético que é derivado do modelo biológico de aprendizado (HOLLAND, 1986). Um classificador genético consiste de uma população de elementos de classificação que competem para fazer a predição. Os elementos que tem o desempenho fraco são retirados e os mais fortes permanecem gerando variações deles próprios.

Os algoritmos genéticos são técnicas utilizadas para solucionar problemas de aprendizado de máquina (DE JONG, 1988), através de tentativa e erro com alguma retro alimentação do erro onde aprendem soluções melhores para os problemas.

Por fim, os sistemas de aprendizado simbólico constroem representações simbólicas de um conceito para aprender, através de análise de exemplos e contra-exemplos desse conceito. As representações simbólicas geralmente estão na forma de expressão lógica, árvores de decisão, regras (ou rede semântica), sendo estas duas últimas as mais estudadas atualmente.

Sistemas como o ID3 (QUINLAN, 1986) e C4.5 (QUINLAN, 1993), que é o algoritmo utilizado neste trabalho, são utilizados para indução de árvores de decisão e contribuíram bastante na pesquisa em Inteligência Artificial. O desenvolvimento original do programa para a indução de árvores de decisão é atribuído a Morgan e Messeger (1973).

2.1.5 Modos de Aprendizado Automático

Para alguns sistemas de aprendizado é necessário predizer se certa ação irá fornecer certa saída (RUSSELL; NORVIG, 2004). Assim é possível classificar os sistemas de aprendizado automático de três modos (OSÓRIO, 1999):

a) Aprendizado Supervisionado: no qual a partir de um conjunto de observações ou exemplos com rótulos, isto é, a classe de cada exemplo é conhecida, tem-se o objetivo de encontrar uma hipótese que consiga classificar novas observações entre as classes existentes.

b) Aprendizado Não-supervisionado ou *clustering*: no qual a partir de um conjunto de observações ou exemplos sem rótulos, o objetivo é estabelecer a existência de *clusters* ou similaridades nesses dados.

c) Aprendizado Semi-Supervisionado: Surgido recentemente, é o aprendizado onde poucos exemplos rotulados são utilizados ao invés de uma quantidade expressiva como a necessária no aprendizado supervisionado. Ele representa a junção dos aprendizados supervisionado e não-supervisionado, permitindo reduzir a necessidade de dados rotulados quando somente um pequeno conjunto de exemplos rotulados está disponível. Também é conhecido como aprendizado por reforço, onde se dispõe apenas de uma avaliação qualitativa do comportamento do sistema, no entanto sem poder medir quantitativamente o erro (desvio do comportamento em relação à referência desejada).

2.1.6 Formas de Aprendizado

Os algoritmos de aprendizado podem ser classificados de duas formas segundo a maneira em que os exemplos são apresentados (SANCHES; GEROMINI, 2001):

a) Não incremental: necessita que todos os exemplos, ao mesmo tempo, estejam disponíveis para que seja induzido um conceito. Estes algoritmos são vantajosos para problemas de aprendizados nos quais todos os exemplos estão disponíveis e não ocorrerão mudanças. Também é conhecido como *batch*. Um exemplo de sistema que originalmente utilizam esta forma de aprendizado são as árvores de indução de regras (ID3 e C4.5).

b) Incremental: revê a definição do conceito corrente, caso necessário, em resposta a cada novo exemplo de treinamento observado. Cada exemplo observado é considerado um a um pelo sistema. Desta maneira o sistema segue modificando o conceito à medida que novos exemplos são inseridos. Um exemplo de sistema incremental são as redes Neurais do tipo incremental, as quais se modificam com novos exemplos.

Uma vantagem do algoritmo incremental é atualizar rapidamente o conhecimento a cada novo exemplo, uma vez que pode ser mais eficiente rever uma hipótese existente do que gerar uma nova hipótese a cada novo exemplo observado.

2.1.7 Linguagens de Descrição (ou Representação)

Todo sistema de aprendizado necessita ter uma linguagem para poder descrever ou representar exemplos, ou objetos (ou possíveis eventos), uma linguagem para descrever hipóteses, ou conceitos e uma para descrever conhecimento de fundo (ou teoria do domínio) (REZENDE, 2002).

a) Linguagem de descrição de Exemplos (Objetos) - LDE:

Descrições Estruturais: Retrata um objeto como sendo uma estrutura complexa de vários componentes além das relações entre eles. Como exemplo, a descrição estrutural de um arco pode ser: Um arco consiste de três componentes – dois postes e uma cobertura, tal que cada um deles é um bloco; os dois postes suportam a cobertura; os postes estão na vertical, são paralelos, não se tocam e a cobertura está na horizontal (REZENDE, 2002).

Descrições de Atributos: Nesta descrição um objeto é descrito através de suas características globais como um vetor de valores de atributos. Um atributo pode ser de qualquer tipo, numérico ou não numérico. Uma descrição de atributos de um arco pode ser: Seu tamanho é seis centímetros, sua altura é três centímetros e sua cor é verde (REZENDE, 2002).

b) Linguagem de descrição de Hipóteses (Conceitos) - LDH:

Formalmente, para descrever conceitos em aprendizado automático utiliza-se:

- As Árvores de decisão,

- As Regras se-então,
- As Redes semânticas,
- A Lógica de predicados.

c) Linguagem de descrição de Conhecimento de Fundo (domínio) – LDC:

É utilizada para descrever algum conhecimento prévio e relevante do domínio do problema. O conhecimento de fundo abrange conceitos absorvidos anteriormente, hipóteses e métodos para avaliá-las, restrições do domínio, metas, entre outros. As linguagens para descrição de conhecimento de fundo são parecidas com as utilizadas para descrição de hipóteses (REZENDE, 2002).

No trabalho a ser desenvolvido não serão utilizadas as linguagens aqui descritas.

2.1.8 Estratégias de Aprendizado Automático

Existem várias estratégias (SANCHES; GEROMINI, 2001) de Aprendizado Automático, as quais se podem citar em ordem de complexidade de inferência apresentada pelo aprendiz: aprendizado por hábito, por instrução, por dedução, por analogia e por indução, sendo esta última a utilizada no sistema proposto neste trabalho.

2.1.8.1 Aprendizado por Indução ou Indutivo

O aprendizado por indução é a estratégia de aprendizado que os seres humanos utilizam ao fazerem generalizações a partir de alguns fatos ou ao obter padrões em grupos de informações que parecem caóticas (SANCHES; GEROMINI, 2001).

A indução se caracteriza pelo raciocínio que parte do específico para o geral.

Nesta estratégia o aprendiz obtém um conceito ao fazer inferências indutivas sobre os fatos apresentados. Entretanto, as hipóteses geradas pela inferência indutiva podem ou não manter a verdade.

A inferência indutiva é um importante meio para se criar novos conhecimentos e prever eventos futuros. A indução permite obter conclusões genéricas sobre um conjunto particular de exemplos. Mas é preciso ter cuidado com as generalizações geradas, uma vez que o conhecimento gerado pode ser de pouco ou nenhum valor.

As hipóteses geradas pela inferência indutiva podem ou não preservar a verdade, ou seja, as hipóteses podem levar a conclusões cujos conteúdos excedam aos das premissas. Por isto os argumentos indutivos são indispensáveis para fundamentação de uma significativa parte dos nossos conhecimentos. Entretanto, este mesmo fato levanta questões complicadas,

que dificultam a análise dos resultados obtidos com o auxílio de métodos indutivos (BATISTA, 2003).

O número de informações colhidas antes de se obter uma regra é um ponto importante. Pode-se observar isto com o exemplo do homem que chega a uma determinada empresa e encontra um faxineiro na entrada principal, depois vai ao banheiro e encontra outro faxineiro fazendo limpeza, a conclusão dele, com somente estas duas amostras, é de que na empresa só existem faxineiros. Neste caso a indução foi realizada com um número insuficiente de exemplos.

Atributos corretos também é um fator importante. Um exemplo é o caso fictício de uma criança brincando com uma aranha (SANCHES; GEROMINI, 2001). A aranha está parada e ela dá um grito, assim a aranha foge. Então ela arranca duas pernas da aranha e dá outro grito, fazendo a aranha fugir novamente. Ela arranca novamente duas pernas e grita novamente, a aranha foge. Até que ao retirar os dois últimos pares de pernas da aranha e gritar, a aranha não se move. A criança conclui que a aranha sem pernas não escuta, portanto utiliza as pernas para ouvir. Isto é o mesmo que se escolher os atributos A1 e A2 para relacionar com certas classes C1 e C2, entretanto estas classes independem dos atributos escolhidos; as classes C1 e C2 tem relação com os atributos A3 e A4, que não aparecem entre os atributos escolhidos para análise. Neste caso a indução será feita sobre dados irrelevantes.

2.1.9 Métodos de Aprendizado Indutivo

Dois métodos muito conhecidos de aprendizado indutivo a partir de exemplos são os aprendizados AQ e Dividir para Conquistar.

2.1.9.1 Aprendizado AQ

O aprendizado AQ utiliza o método de progressivamente cobrir os dados de treinamento enquanto as regras de decisão são geradas. Seu funcionamento se baseia na busca de um conjunto de regras (conjunções de pares atributo-valor ou predicados arbitrários) que cubram todos os exemplos positivos e nenhum exemplo negativo. O método AQ generaliza, passo a passo, as descrições dos exemplos positivos selecionados (MICHALSKY; BRATKO; KUBAT, 1998).

2.1.9.2 Aprendizado Dividir para Conquistar

O método Dividir para Conquistar divide o conjunto de exemplos em subconjuntos, nos quais se pode trabalhar com facilidade, em seguida recursivamente a mesma estratégia é aplicada a cada subconjunto. Dentro de este tipo de aprendizado, encontramos a família TDIDT (*Top-Down Induction Trees*), na qual está o algoritmo de aprendizado a ser utilizado neste trabalho.

2.2 Algoritmo C4.5

O algoritmo C4.5 (QUINLAN, 1993) é um sistema de aprendizado que constrói árvores de decisão a partir de um conjunto de exemplos. Ele é o algoritmo utilizado neste trabalho para gerar regras. O C4.5 faz parte da família TDIDT, a qual pertence aos métodos indutivos de aprendizado automático que aprendem a partir de exemplos pré-classificados.

Com base no que foi descrito nas seções anterior o C4.5 está classificado como um sistema de aprendizado indutivo, supervisionado e não incremental.

2.2.1 Indução de Árvores de Decisão

Em (UTGOFF, 1989) é feita uma definição formal de árvores de decisão, como sendo uma estrutura de dados composta por n (n>1) nós. Estes nós podem ser:

- Um nó folha, ou nó resposta, que contém um nome de classe, ou
- Um nó raiz, ou nó de decisão, que contém um atributo de teste, sendo que para cada um dos possíveis valores deste atributo existe um ramo para uma outra árvore de decisão.

Na figura 2.1 extraída de (REZENDE, 2002) é apresentado um exemplo de árvore de decisão para o diagnóstico de um paciente. Cada elipse representa um teste em um atributo para um dado conjunto de dados de pacientes, os retângulos representam uma classe, no caso do exemplo, um diagnóstico. Para fazer o diagnóstico em um paciente é necessário iniciar na raiz e seguir cada teste até alcançar uma folha.

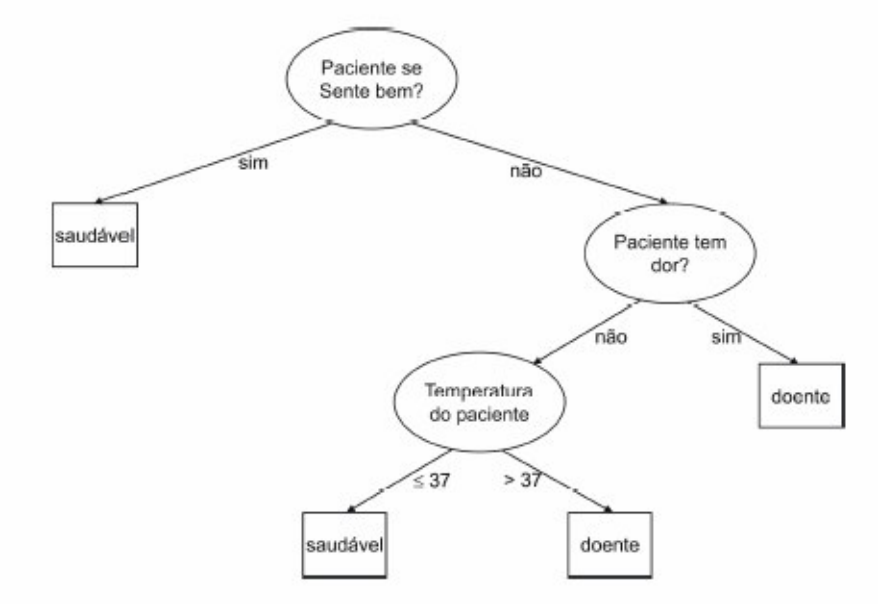


Figura 2.1 – Árvore de decisão para diagnóstico em paciente (REZENDE, 2002).

A árvore pode ser representada por regras. As regras têm seu início na raiz e terminam em uma das folhas da árvore. Na árvore da figura 2.1, pode-se identificar a primeira regra como sendo: SE paciente se sente bem = sim ENTÃO classe = saudável.

2.2.2 Construção de Árvores de Decisão

A construção de uma árvore de decisão é feita pelo método de HUNT, MARIN e STONE (1966). A partir de um conjunto de treinamento *T*, sejam as classes denotadas por $\{C_1, C_2, \ldots, C_k\}$, os seguintes passos devem ser seguidos (SERVENTE, 2002):

1. *T* contém um ou mais casos, todos pertencentes a uma única classe C_j. Nesse caso, a árvore de decisão para *T* é uma folha identificando a classe C_j;

2. *T* não contém nenhum caso. Neste caso, a árvore é uma folha, mas a classe associada deve ser determinada pela de informação que não pertence a *T*. Por exemplo, a classe mais freqüente para o nó pai desse nó pode ser utilizada;

3. *T* contêm casos pertencentes a várias classes. Nesse caso, a idéia é refinar *T* em subconjuntos de casos que tendam, ou aparentam tender a uma coleção de casos pertencentes a uma única classe. Escolhe-se um teste baseado em um único atributo que tem um ou mais resultados mutuamente excludentes $\{O_1, O_2, \ldots, O_n\}$. *T* se divide nos subconjuntos $T_1, T_2, \ldots, T_n$, onde T_i contém todos os casos de *T* que tem o resultado O_i para a partição escolhida. A árvore de decisão para *T* consiste em um nó de decisão identificado pela partição, com um ramo para cada um dos resultados possíveis.

Este mecanismo de construção é aplicado recursivamente para cada subconjunto de casos de treinamento, de maneira que o iésimo ramo forme a árvore de decisão construída pelo subconjunto T_i, de dados de treinamento.

2.2.2.1 Cálculo do Ganho de Informação

Quando o conjunto *T* contém exemplos pertencentes a distintas classes, realiza-se um cálculo sobre os distintos atributos e uma partição segundo o melhor atributo. Através da teoria da informação encontra-se o melhor atributo. A teoria sustenta que a informação se maximiza com a minimização da entropia. A entropia é uma medida de incerteza de um conjunto, ou a medida da impureza de uma coleção arbitrária de exemplos. A entropia assume o valor máximo (1) quando existem tantos elementos positivos como negativos, e o valor mínimo (0) quando todos os elementos são da mesma classe.

Entropia(S_i) é igual ao número esperado de bits necessários para codificar a classe ("+" ou "-") de um elemento de *S* escolhido aleatoriamente com distribuição uniforme.

Da Teoria da Informação, um código de comprimento ótimo atribui $-\log_2 p$ bits a uma mensagem que tenha probabilidade *p*. Logo, o número esperado de bits para codificar "+" ou "-" de elementos selecionados aleatoriamente de S é a probabilidade p^+ multiplicado por $-\log_2 p^+$ mais a probabilidade p^- multiplicado por $-\log_2 p^-$.

As equações apresentadas nesta seção estão desenvolvidas conforme (QUINLAN, 1993).

Em um subconjunto S_i com exemplos positivos e negativos, o cálculo da entropia $H(S_i)$ é dado por:

$$H(S_i) = -p_i^+ \log_2 p_i^+ - p_i^- \log_2 p_i^- \tag{2.1}$$

Onde P_i^+ é a probabilidade de um exemplo de S_i, tomado ao acaso, seja positivo. S_i contém n_i^+ exemplos positivos e n_i^- exemplos negativos. Esta probabilidade é calculada por:

$$p_i^+ = \frac{n_i^+}{n_i^+ + n_i^-} \tag{2.2}$$

Analogamente a P_i^+ calcula-se P_i^-, porém trocando a quantidade de exemplos positivos pela de exemplos negativos e vice-versa.

Se houverem mais de duas classes (+ e -), ou seja, n classes, a entropia é dada por:

$$H(S_i) = -p_i^1 \log_2 p_i^1 - p_i^2 \log_2 p_i^2 - p_i^3 \log_2 p_i^3 \ldots\ldots - p_i^n \log_2 p_i^n \tag{2.3}$$

Generalizando a expressão $H(S_i)$ para qualquer tipo de exemplos tem-se a expressão geral da entropia:

$$H(S_i) = \sum_{i=1}^{n} - p_i \log_2 p_i \tag{2.4}$$

Quando um atributo *at* divide o conjunto *S* em subconjuntos S_i, i = 1,2,....., n, a entropia total dos subconjuntos é a somatória das multiplicações da probabilidade ($P(S_i)$) de que um exemplo pertença a S_i pela entropia ($H(S_i)$) do subconjunto S_i :

$$H(S, at) = \sum_{i=1}^{n} P(S_i) \cdot H(S_i) \tag{2.5}$$

O ganho da informação pode ser calculado como a diminuição em entropia:

$$I(S, at) = H(S) - H(S, at) \tag{2.6}$$

Sendo $H(S)$ o valor da entropia antes de realizar a subdivisão e $H(S,at)$ o valor da entropia do sistema de subconjunto gerados pela partição segundo *at*.

2.2.2.2 Divisão dos dados

O método Dividir para conquistar divide os dados do nó através de uma partição realizada sobre o melhor atributo. O processo de construção da árvore não deve somente encontrar qualquer partição que gere subconjuntos com uma única classe cada; o necessário é encontrar uma árvore que revele uma estrutura do domínio, portanto tendo poder preditivo. A partição deve ter a menor quantidade de classes possíveis. No caso ideal, escolher a cada passo a partição que gere a árvore menor.

2.2.2.2.1 Critério do Ganho

Imagine que existe uma partição possível com *n* resultados que dividem o conjunto de treinamento *T* nos subconjuntos $T_1, T_2, \ldots, T_n$. Se a partição for realizada sem avaliar as divisões posteriores dos subconjuntos T_i, a única informação disponível para avaliar a partição é a distribuição de classes em *T* e seus subconjuntos.

Considerando um valor similar, com *T* dividido em *n* resultados da partição *X*. A entropia pode se calculada como a soma ponderada dos subconjuntos (QUINLAN, 1995):

$$H(T,X)=\sum_{i=1}^{n}\frac{|T_i|}{|T|}\times H(T_i) \tag{2.7}$$

O ganho da informação a partir de *T* de acordo com a partição *X* é:

$$I(T,X)=H(T)-H(T,X) \tag{2.8}$$

O critério de ganho seleciona a partição que maximiza o ganho de informação. Nele se calcula o ganho que resultaria de dividir o conjunto de dados de acordo com cada atributo possível antes de dividir os dados em cada nó. A partição escolhida é aquela que resulta no maior ganho.

2.2.2.2.2 Critério da Proporção do Ganho

Realizando uma partição sobre um atributo que seja o nó primário de um conjunto de dados, se obtém um único subconjunto para cada caso, para cada subconjunto teremos *I(T,X)=0*, assim o ganho de informação será máximo. Este tipo de partição não é útil.

Este fato que acontece no critério de ganho é corrigido através de uma normalização, na qual se ajusta o ganho aparente. Tomando-se o conteúdo de informação de uma mensagem correspondente aos resultados das divisões. Por analogia à definição de *I(S)* temos (QUINLAN, 1995):

$$I\_divisão\,(X)=-\sum_{i=1}^{n}\frac{|T_i|}{|T|}\times\log_2\left(\frac{|T_i|}{|T|}\right) \tag{2.9}$$

Isto representa a informação potencial gerada ao dividir *T* em *n* subconjuntos, enquanto que o ganho da informação mede a informação relevante a uma classificação que nasce da mesma partição (QUINLAN, 1995).

$$Proporção\_de\_ganho(X)=\frac{I(T,X)}{I\_divisão(X)} \tag{2.10}$$

A expressão *Proporção_de_ganho(X)* calcula a proporção útil de informação gerada na divisão. Se a divisão é normal a informação da divisão será pequena e esta proporção se tornará instável. O critério de proporção de ganho elege uma divisão que maximiza a expressão.

2.2.3 Detalhamento do Algoritmo C4.5

O pseudo-código do algoritmo C4.5 para construir árvores de decisão a partir de um conjunto de exemplos é apresentado na figura 2.2.

```
Função C4.5
(R: conjunto de atributos não classificadores,
C: atributo classificador,
S: conjunto de treinamento) devolve uma árvore de decisão;

Início
Se S está vazio,
          devolver um único nó com Valor Falha;
Se todos os registros de S têm o mesmo valor para o atributo classificador,
          devolver um único nó com tal valor;
Se R está vazio, então
          devolver um único nó com o valor mais freqüente do atributo
          classificador nos registros de S [Nota: existirão erros, isto é,
          registros que não estarão bem classificados neste caso];
Se R não está vazio, então
          D←atributo com maior Proporção de Ganho(D,S) entre os atributos de R;
          Sejam {dj | j=1,2, .., m} os valores do atributo D;
          Sejam {Sj | j=1,2, .., m} os subconjuntos de S correspondentes aos
          valores de dj respectivamente;
          devolver uma árvore com a raiz nomeada como D e com os arcos nomeados
          d1, d2, .., dm que vão respectivamente às árvores
          C4.5(R-{D}, C, S1), C4.5(R-{D}, C, S2), .., C4.5(R-{D}, C, Sm);
Fim
```

Figura 2.2 – Algoritmo C4.5 (QUINLAN, 1993).

A seguir são descritas as características, a poda em árvores de decisão, a construção de uma árvore de decisão e a geração de regras de decisão, tendo como foco o algoritmo C4.5.

2.2.3.1 Características

O C4.5 tem como função criar uma descrição de um conjunto de dados através de uma árvore de decisão. A partir de dados sem contradição entre eles, a árvore gerada descreverá o conjunto de dados de entrada com perfeição (QUINLAN, 1993). Dados:

- Um conjunto de dados;
- Um conjunto de descritores (atributos) de cada dado;
- Um classificador (classe)/conjunto de classificadores para cada objeto.

Deseja-se obter uma árvore onde os nós podem ser:

1. Nós intermediários: onde se encontram os descritores escolhidos segundo os critérios de entropia e ganho ou proporção de ganho, que determinam qual ramo é o que deve tomar-se.

2. Folhas: estes nós determinam o valor do classificador.

Não devem existir dois objetos pertencentes a diferentes classes com valores idênticos para cada um de seus atributos, se isto acontecer, os atributos são inadequados para o processo de classificação.

Diferentes provas são feitas para dividir os dados em cada nó. As provas possíveis são:

1. Prova padrão para os atributos discretos, com um resultado e um ramo para cada valor possível do atributo.
2. Uma prova mais complexa, baseada em um atributo discreto, onde os valores possíveis são designados a um número variável de grupos com um resultado possível para cada grupo, em lugar de cada valor do atributo.
3. Se um atributo *A* tem valores numéricos contínuos, se realiza uma prova binária com resultados $A \leq Z$ *e* $A > Z$, para o qual se deve determinar o valor limite *Z*.

Estas provas são avaliadas da mesma maneira, observando o resultado da proporção de ganho ou alternativamente o ganho, resultante da divisão que produzem.

O método utilizado (QUINLAN, 1993) para provas sobre atributos contínuos é simples, primeiro os exemplos de treinamento *T* são ordenados segundo os valores do atributo *A* contínuo que está sendo considerado. Existe um número finito destes valores. Sendo $\{v_1, v_2, \ldots, v_m\}$ os valores possíveis do atributo *A*. Qualquer valor limite entre v_i e v_{i+1} terá o mesmo efeito ao dividir os exemplos entre aqueles cujo valor para *A* pertence ao subconjunto $\{v_1, v_2, \ldots, v_i\}$ e aqueles cujo valor pertence a $\{v_{i+1}, v_{i+2}, \ldots, v_m\}$. Assim, existem $m - 1$ divisões possíveis segundo o valor de *A* e todas são examinadas. Depois de ordenados, as provas para todos os valores podem ser realizadas de uma única vez.

Para atributos desconhecidos o C4.5 assume que todos os resultados de provas desconhecidos distribuem-se probabilisticamente segundo a freqüência relativa dos valores conhecidos. Um caso com um valor desconhecido se divide em fragmentos cujos pesos são proporcionais às freqüências relativas, dando por resultado que um caso pode seguir múltiplos caminhos na árvore. Isto se aplica tanto quando os casos de treinamento dividem-se durante a construção da árvore como quando a árvore é utilizada para classificar casos.

2.2.3.2 Poda das Árvores de Decisão

O método recursivo de divisão dos dados para construção das árvores de decisão, dividirá o conjunto de treinamento até que um subconjunto tenha somente uma classe, ou até

que a prova não ofereça melhora. Em função disto normalmente a árvore resultante fica muito mais complexa que a necessária para descrever o conjunto de exemplos (QUINLAN, 1995).

É necessário podar a árvore, porém não é simplesmente apagando a árvore em favor de um ramo que se simplifica a mesma, mas sim eliminando as partes da árvore que não contribuem para a correta classificação de novos casos, produzindo assim uma árvore menos complexa e mais compreensível.

Duas maneiras de modificar o método de divisão recursivo para produzir árvores mais simples são decidir não dividir mais o conjunto de treinamento ou remover retrospectivamente alguma parte da estrutura construída pela divisão recursiva.

O C4.5 realiza a remoção de partes da árvore depois da mesma estar produzida. O crescimento e posterior poda das árvores são mais lentos, mas mais confiáveis. Podar a árvore totalmente produzida pode ser computacionalmente ineficiente, no sentido que não é usual encontrar exemplos de domínios onde uma árvore extremamente grande (com milhares de nós, por exemplo) é pós-podada em poucas centenas de nós. A alternativa de parada no procedimento de crescimento da árvore, tão logo a divisão seja considerada não confiável, também apresenta um ponto negativo no seu algoritmo, corre-se o risco de selecionar uma árvore sub-ótima ao interromper o crescimento da árvore (BREIMAN, 1984).

A poda é feita com base na proporção de erros (QUINLAN, 1993), onde se começa pelas folhas e se examina cada sub-árvore, se uma troca de uma sub-árvore por uma folha ou ramo leva a uma menor proporção de erros, então se deve podar a árvore, recordando que as proporções de erros já calculadas para todas as sub-árvores serão afetadas. Como a proporção de erros predita de uma árvore diminui se as proporções de erros preditas de seus ramos diminuem, então é gerada uma árvore com proporção de erro mínima.

Para estimar a proporção de erros, quando uma folha cobre N casos de treinamento, sendo E deles de forma errônea, o estimador da proporção de erros de substituição para esta folha é N/E. Podemos entender isto como E eventos em N provas.

Sendo N a quantidade de casos de treinamento cobertos por uma folha e E a quantidade de erros preditos se um conjunto de N novos casos forem classificados pela árvore. A soma dos erros preditos nas folhas, dividido pelo número de casos de treinamento é um estimador imediato do erro de uma árvore podada sobre novos casos.

2.2.3.3 Construção de uma Árvore de Decisão utilizando o C4.5

A seguir será apresentado um exemplo de construção de árvore de decisão, extraído de (SERVENTE, 2002).

A partir dos seguintes dados:

Tabela 2.1 – Dados do exemplo Joga Tênis (SERVENTE, 2002).

Estado	**Umidade**	**Vento**	**Joga Tênis**
?	Alta	Leve	Não
Ensolarado	Alta	Forte	Não
Nublado	Alta	Leve	Sim
Chuva	Alta	Leve	Sim
Chuva	Normal	Leve	Sim
Chuva	Normal	Forte	Não
Nublado	Normal	Forte	Sim
Ensolarado	Alta	Leve	Não
Ensolarado	Normal	Leve	Sim
Chuva	Normal	Leve	Sim
Ensolarado	Normal	Forte	Sim
Nublado	Alta	Forte	Sim
Nublado	Normal	Leve	Sim
Chuva	Alta	Forte	Sim

Neste exemplo a distribuição de dados para o atributo Estado é:

Tabela 2.2 – Distribuição de Dados para o atributo Estado (SERVENTE, 2002).

	Desconhecido	Ensolarado	Nublado	Chuva
Não	1	2	0	1
Sim	0	2	4	4
Total	1	4	4	5

Inicialmente é feito o cálculo da entropia do conjunto, para isto os atributos desconhecidos não são considerados. Portanto, no exemplo existe um total de 13 casos, dos quais 3 são positivos. Tem-se,

$$H(S) = -\frac{3}{13}\log_2\frac{3}{13} - \frac{10}{13}\log_2\frac{10}{13} = 0.7793 bits$$

Calcula-se em seguida a entropia que os conjuntos resultantes da divisão de dados segundo este atributo teriam.

$$H(S, Estado) = \frac{4}{13}\left(-\frac{2}{4}\log_2\frac{2}{4} - \frac{2}{4}\log_2\frac{2}{4}\right) + \frac{4}{13}\left(-\frac{0}{4}\log_2\frac{0}{4} - \frac{4}{4}\log_2\frac{4}{4}\right) + \frac{5}{13}\left(-\frac{1}{5}\log_2\frac{1}{5} - \frac{4}{5}\log_2\frac{4}{5}\right) = 0.58536 bits$$

O próximo cálculo é o ganho resultante da divisão do subconjunto segundo o atributo Estado:

$$Ganho\ (S, Estado) = \frac{13}{14}(0.7793 - 0.58536) = 0.180 bits$$

Ao calcular a informação da divisão é necessário ter em conta uma categoria extra para o valor desconhecido para o atributo. A informação da divisão calcula-se como:

$$I\_divisão\ (S) = -\frac{4}{14}\times\log_2\left(\frac{4}{14}\right) - \frac{4}{14}\times\log_2\left(\frac{4}{14}\right) - \frac{5}{14}\times\log_2\left(\frac{5}{14}\right) - \frac{1}{14}\times\log_2\left(\frac{1}{14}\right) = 1.835 bits$$

Enfim, calcula-se a proporção de ganho.

$$Proporção\_de\_ganho\ (S) = \frac{Ganho\ (S)}{I\_divisão\ (S)} = 0.098 bits$$

Utiliza-se o mesmo cálculo do ganho e da proporção de ganho para o atributo Umidade, obtendo-se os valores:

Ganho = 0.0746702 bits Proporção de ganho = 0.0746702 bits

Para o atributo Vento obtêm-se os seguintes valores:

Ganho = 0.00597769 bits Proporção de ganho = 0.0060687 bits

Tanto pelos valores do ganho como proporção de ganho é conveniente dividir o conjunto segundo o atributo Estado. Ao dividir os 14 casos para continuar com a construção da árvore, os 13 casos para os que o valor de Estado é conhecido, não apresentam problemas e são repartidos segundo o valor do Estado. O caso que não se conhece o valor do Estado é repartido entre os conjuntos que contém Ensolarado, Nublado e Chuva com os pesos 4/13, 4/13 e 5/13 respectivamente.

Tendo como exemplo a divisão dos dados para o valor Nublado do atributo Estado, teremos o seguinte subconjunto:

Tabela 2.3 – Subconjunto com valor Nublado do atributo Estado (SERVENTE, 2002).

Estado	**Umidade**	**Vento**	**Joga Tênis**	**Peso**
?	Alta	Leve	Não	4/13
Nublado	Alta	Leve	Sim	1
Nublado	Normal	Forte	Sim	1
Nublado	Alta	Forte	Sim	1
Nublado	Normal	Leve	Sim	1

A distribuição de dados para o atributo Umidade é:

Tabela 2.4 – Distribuição de Dados para o atributo Estado com valor Umidade (SERVENTE, 2002).

	Desconhecido	Alta	Normal
Não	0	0.3	0
Sim	0	2	2
Total	0	2.3	2

Para o atributo Umidade, obtêm-se os valores:

Ganho = 0.068 bits Proporção de ganho = 0.068 bits

Para o atributo Vento obtêm-se os seguintes valores:

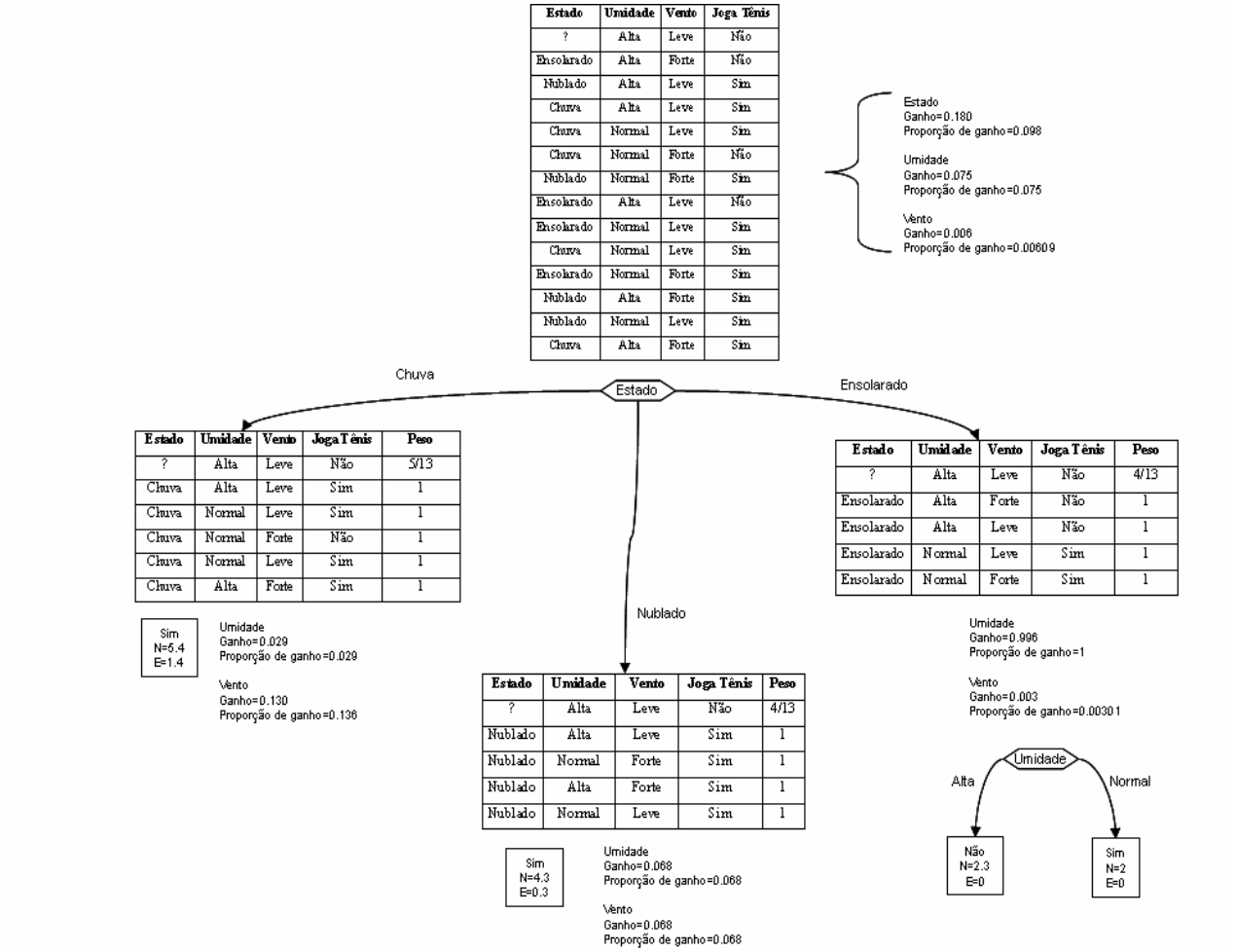
Ganho = 0.068 bits Proporção de ganho = 0.068 bits

Neste caso vemos que a divisão do conjunto de dados não oferece nenhuma melhora, portanto, colapsamos a árvore para a folha Sim, que é a que maior peso tem. A quantidade de casos N cobertos pela folha é 4.3 e a quantidade de casos cobertos incorretamente E associado a folha é 0.3.

Na figura 2.4 é apresentado um esquema dos passos para a construção da árvore de decisão deste exemplo. A seguir tem-se a árvore obtida:

```
Estado = Nublado: Sim (4.3/0.3)
Estado = Chuva: Sim (5.4/1.4)
Estado = Ensolarado:
Umidade = Alta: Não (2.3)
Umidade = Normal: Sim (2.0)
```

Figura 2.3 – Árvore obtida no exemplo (SERVENTE, 2002).



Estado	Umidade	Vento	Joga Tênis
?	Alta	Leve	Não
Ensolarado	Alta	Forte	Não
Nublado	Alta	Leve	Sim
Chuva	Alta	Leve	Sim
Chuva	Normal	Leve	Sim
Chuva	Normal	Forte	Não
Nublado	Normal	Forte	Sim
Ensolarado	Alta	Leve	Não
Ensolarado	Normal	Leve	Sim
Chuva	Normal	Leve	Sim
Ensolarado	Normal	Forte	Sim
Nublado	Alta	Forte	Sim
Nublado	Normal	Leve	Sim
Chuva	Alta	Forte	Sim

Estado	Umidade	Vento	Joga Tênis	Peso
?	Alta	Leve	Não	5/13
Chuva	Alta	Leve	Sim	1
Chuva	Normal	Leve	Sim	1
Chuva	Normal	Forte	Não	1
Chuva	Normal	Leve	Sim	1
Chuva	Alta	Forte	Sim	1

Estado	Umidade	Vento	Joga Tênis	Peso
?	Alta	Leve	Não	4/13
Ensolarado	Alta	Forte	Não	1
Ensolarado	Alta	Leve	Não	1
Ensolarado	Normal	Leve	Sim	1
Ensolarado	Normal	Forte	Sim	1

Estado	Umidade	Vento	Joga Tênis	Peso
?	Alta	Leve	Não	4/13
Nublado	Alta	Leve	Sim	1
Nublado	Normal	Forte	Sim	1
Nublado	Alta	Forte	Sim	1
Nublado	Normal	Leve	Sim	1

Figura 2.4 – Construção de uma árvore de decisão utilizando C4.5 (SERVENTE, 2002).

2.2.3.4 Geração de Regras de Decisão utilizando o C4.5

No exemplo em análise o C4.5 utiliza a árvore gerada sem simplificação e constrói uma regra de decisão para cada folha da árvore. As regras geradas neste caso são:

```
Regra 1
SE Estado = Ensolarado
E Umidade = Alta
ENTÃO JogoTênis = Não
Regra 2
SE Estado = Ensolarado
E Umidade = Normal
ENTÃO JogoTênis = Sim
Regra 3
SE Estado = Nublado
ENTÃO JogoTênis = Sim
Regra 4
SE Estado = Chuva
ENTÃO JogoTênis = Sim
```

Figura 2.5 – Regras geradas no exemplo (SERVENTE, 2002).

Em seguida o C4.5 tenta generalizar as regras, para isto ele elimina as condições que geram uma maior estimativa pessimista de erro.

Para estimar a probabilidade de erro, buscam-se limites de confiança CF. O limite superior desta probabilidade pode ser obtido a partir dos limites de confiança para a distribuição binomial e é escrito como $U_{CF}(E, N)$ (E erros entre N exemplos classificados). Os limites superior e inferior são simétricos na distribuição binomial, portanto a probabilidade de que a média real de erros exceda $U_{CF}(E, N)$ é CF/2. A estimativa de erro pessimista de classificação de N exemplos é $N \times U_{CF}(E,N)$ (QUINLAN, 1993).

A estimativa de erro será calculada para as regras geradas e para as regras resultantes da eliminação de cada uma de suas condições.

Na primeira regra temos:

Tabela 2.5 – Estimativas de erro para a Regra 1 (SERVENTE, 2002).

Erros	Quant. de casos cobertos	Estimativa pessimista do erro	Condição ausente
0	2	50%	<regra atual>
4	7	75.5%	Estado=Ensolarado
2	4	77.1%	Umidade=Alta

A retirada de qualquer condição possui uma estimativa pessimista de erro superior a da regra atual, portanto não é interessante eliminar nenhuma condição, assim a regra atual é mantida.

```
Regra 1
SE Estado = Ensolarado
E Umidade = Alta
ENTÃO JogoTênis = Não [50%]
```

Figura 2.6 – Regra 1 do exemplo (SERVENTE, 2002).

Todas as outras regras devem ser avaliadas. Para a segunda regra tem-se:

Tabela 2.6 – Estimativas de erro para a Regra 2 (SERVENTE, 2002).

Erros	Quant. de casos cobertos	Estimativa pessimista do erro	Condição ausente
0	2	50%	<regra atual>
1	7	33.8%	Estado=Ensolarado
2	4	77.1%	Umidade=Normal

Neste caso, a retirada da condição Estado apresenta uma menor estimativa pessimista de erro, assim a condição deve ser retirada e o cálculo refeito:

Tabela 2.7 – Estimativas de erro para a Regra 2, retirada condição Estado (SERVENTE, 2002).

Erros	Quant. de casos cobertos	Estimativa pessimista do erro	Condição ausente
1	7	33.8%	<regra atual>
4	14	41.3%	Umidade=Normal

```
Regra 2
SE Estado = Ensolarado
E Umidade = Normal
ENTÃO JogoTênis = Sim [66.2%]
```

Figura 2.7 – Regra 2 do exemplo (SERVENTE, 2002).

Para a terceira regra temos:

Tabela 2.8 – Estimativas de erro para a Regra 3 (SERVENTE, 2002).

Erros	Quant. de casos cobertos	Estimativa pessimista do erro	Condição ausente
0	4	29.3%	<regra atual>
4	14	41.3%	Estado=Nublado

```
Regra 3
SE Estado = Nublado
ENTÃO JogoTênis = Sim [70.7%]
```

Figura 2.8 – Regra 3 do exemplo (SERVENTE, 2002).

Para a quarta regra temos:

Tabela 2.9 – Estimativas de erro para a Regra 4 (SERVENTE, 2002).

Erros	Quant. de casos cobertos	Estimativa pessimista do erro	Condição ausente
1	5	45.4%	<regra atual>
4	14	41.3%	Estado=Chuva

```
Regra 4
SE Estado = Chuva
ENTÃO JogoTênis = Sim [54.6%]
```

Figura 2.9 – Regra 4 do exemplo (SERVENTE, 2002).

Depois que as regras são generalizadas, as mesmas são agrupadas por classe e determinam-se os subconjuntos de regras que geram uma codificação mínima para a classe. O C4.5 calcula para cada subconjunto de regras a quantidade de bits necessários para codificar as regras e determinar quais regras são convenientes utilizar para representar cada classe.

No exemplo, a regra 1 para a classe Não e a regra 3 para a classe Sim são as escolhidas.

O último passo é escolher a classe padrão. Para isto calcula-se a quantidade de casos de cada classe não cobertos pelas regras escolhidas:

Tabela 2.10 – Casos coberto e não coberto por classe (SERVENTE, 2002).

Casos	Não Cobertos	Classe
4	2	Não
10	6	Sim

O maior número de casos não cobertos está na classe Sim, portanto sendo está a classe padrão. O conjunto final de regras é:

```
Regra 1
SE Estado = Ensolarado
E Umidade = Alta
ENTÃO JogoTênis = Não [50.0%]
Regra 3
SE Estado = Nublado
ENTÃO JogoTênis = Sim [70.7%]
Regra 5
Classe padrão = Sim
```

Figura 2.10 – Conjunto final de regras do exemplo (SERVENTE, 2002).

O exemplo apresentado finaliza o estudo do algoritmo C4.5 e assim o capitulo sobre Aprendizado de Máquina, o capitulo posterior abordará o tema Domótica e um sistema de Domótica Inteligente, o Sistema ABC.

3 DOMÓTICA

A palavra Domótica é a junção da palavra latina Domus (casa) e do termo Robótica (ANGEL, 1993). O significado está relacionado à instalação de tecnologia em residências, principalmente através de dispositivos eletrônicos e eletroeletrônicos, com o objetivo de melhorar a qualidade de vida, aumentar a segurança e viabilizar o uso racional dos recursos para seus habitantes. Existem outras denominações para a Domótica, entre elas estão "Edifício Inteligente", "Casa Inteligente", "Ambiente Inteligente", entre outros.

Domótica é um novo domínio de aplicação tecnológica, tendo como objetivo básico melhorar a qualidade de vida, reduzindo o trabalho doméstico, aumentando o bem estar e a segurança de seus habitantes e visando também uma utilização racional e planejada dos diversos recursos (ANGEL, 1993).

3.1 Sistemas Domóticos

Para gerenciar de maneira eficiente os diversos dispositivos e atuadores de uma residência, muitos dados devem ser computados e várias tarefas complexas são executadas.

Uma residência inteligente contém um sistema para gerenciar todo tráfego de informação, bem como um sistema de controle dos equipamentos, permitindo um maior conforto com menor gasto de energia (BOLZANI, 2004b).

O sistema que integra todos os dispositivos para automatizar e controlar uma residência é o sistema domótico. Tal sistema é composto por vários elementos, entre eles os atuadores, sensores, controlador, rede de dados ou rede domótica e interface com o usuário.

Os primeiros sistemas domóticos eram pouco flexíveis, caros, não seguiam padronizações, basicamente utilizavam sensores (dispositivos que transformam parâmetros físicos como temperatura, umidade, entre outros, em sinais elétricos apropriados para que os sistemas domóticos possam analisá-los) e atuadores (são dispositivos eletro-mecânicos que têm suas características alteradas conforme os impulsos elétricos recebidos) ligados a um controlador (BRETERNITZ, 2001).

Desenvolver os diversos controles de uma residência em um só sistema é uma tarefa difícil. Por este motivo, a gestão da residência é dividida em vários subsistemas responsáveis cada um por operações bem específicas, os quais são gerenciados por um controlador central.

Os vários subsistemas de um sistema domótico atuam cada qual especificamente em um campo de controle. Os principais campos são (BOLZANI, 2004a):

- Energia Elétrica – monitora a energia da residência e atua com alternativas na falta da mesma.
- Ar-condicionado – controla a temperatura e ventilação da residência.
- Iluminação – controla as lâmpadas, atuando também na diminuição de consumo de energia.
- Segurança e alarme – monitora a intrusão da residência.
- Combate a incêndio – monitora a presença de fumaça e fogo.
- Multimídia – controla funções de vídeo e áudio.
- Água e dejetos – monitora o abastecimento de água, os dejetos e lixo.
- Interface do usuário – atua na interação com o usuário, permite entrada e saída de informações.
- Controle de Acessos – controla entradas (portas e portões).

Atualmente os sistemas são informatizados e computadorizados.

3.1.1 Atuadores e Sensores

Atuadores e sensores são importantes dispositivos de um sistema domótico, eles podem ser interligados diretamente ao controlador ou serem conectados por meio de alguma interface a uma rede de dados para que possam ser utilizados pelo sistema domótico. Suas características são importantes, uma vez que deles dependem diretamente as capacidades de uma residência inteligente.

Existe uma variedade muito grande de atuadores e sensores que possibilitam a execução de ações e o monitoramento de inúmeras grandezas físicas e eventos.

A seguir é listada uma série de atuadores e sensores, utilizados em sistemas domóticos, bem como suas funções (BOLZANI, 2004a):

- Motores de passo e motores miniatura: variam a posição angular.
- Solenóides, hidráulicos e pneumáticos: variam a posição linear.
- Mini-bombas de circulação: escoamento de líquidos.
- Células Peltier: arrefecimento ou aquecimento de superfícies.
- Folhas aquecedoras: aquecimento de superfícies.
- Sensor de Temperatura: fornecem a medida instantânea da temperatura.
- Termovelocimétricos: detectam a velocidade de variação da temperatura no tempo.

- Sensor de Umidade relativa: sensores do tipo capacitivo que fornecem a medida instantânea da umidade relativa sob a forma de sinais analógicos ou barramento de dados proprietário.
- Sensor de Qualidade de ar: medem o nível de CO_2 existente num ambiente.
- Detector magnético de abertura: utilizados normalmente nas portas e janelas a serem controladas.
- Sensor de Intensidade de iluminação: tem como elemento principal o *LDR* (*Light Dependent Resistor*) cuja resistência é função da intensidade de iluminação que nele incide.
- Sensor de Pressão e força: os sensores de força são compostos por um dispositivo principal que tem sua resistência elétrica alterada conforme a força aplicada na membrana.
- Tacométricos: são sensores de velocidade angular.
- Detector de Fumaça: fornecem uma informação digital (sim ou não) da presença de fumaça em um ambiente.
- Detector de Gás: fornecem informação digital da presença de gás (butano ou propano).
- Detector de Movimento: existem vários tipos, como por exemplo, os infravermelhos ativos que emitem feixes infravermelhos entre dois pontos lineares (um transmissor e um receptor) e ao serem interrompidos acionam o alarme, e o PIR (Infra Vermelho Passivo) que detecta o movimento de fontes de calor tais como o corpo humano.
- Detectores Sísmicos ou de vibração: trata-se de dispositivos piezo-elétricos, normalmente cerâmicos, que geram tensão em seus terminais quando sujeitos à aceleração segundo um dado eixo.
- Detectores de chama: estes sensores são baseados em um dispositivo opto eletrônico sensível a uma determinada radiação eletromagnética, infravermelha ou ultravioleta.
- Detectores de nível: são compostos por bóias que acionam interruptores ou relés quando um determinado nível de líquido é atingido.
- Sensor acústico: utilizados para detectar a quebra de vidros de janelas ou portas.

- Botões de pânico: são detectores de toque, com fio ou sem fio, cujo objetivo principal é de alertar sobre a existência de algum evento anormal num ambiente.
- Sensores diversos: qualidade da água, oxigênio dissolvido, condutividade de líquidos e sólidos, salinidade, sensores de componentes químicos, de posicionamento (GPS - *Global Positioning System*), entre outros.

Existem outros atuadores e sensores, os quais são utilizados para fins específicos e não serão apresentados neste documento. Além destes, outros elementos são importantes nos sistemas domóticos e serão detalhados nas próximas seções.

3.1.2 Controlador

O Controlador é o elemento central que gerencia o sistema domótico. Nele reside toda a inteligência do sistema. Normalmente todos os outros elementos do sistema se conectam ao controlador, enviando e recebendo informações.

No controlador chegam as informações dos sensores, as quais são processadas para gerar alguma ação nos atuadores, ou ainda para apresentar alguma informação ao usuário, ou acionar algum elemento como uma sirene ou indicação luminosa.

O controlador pode ter as interfaces de usuário, as quais são necessárias para apresentar e receber informação (via teclado, monitor, entre outros.).

3.1.3 Redes Domóticas

A rede domótica é o barramento (ou cabeamento) que permite realizar a comunicação entre os diferentes dispositivos existentes no sistema domótico. As redes se baseiam em aplicações, onde uma rede separada e independente é utilizada para cada função, portanto podem existir redes destinadas à segurança, à detecção de incêndios, ao controle de acessos, à climatização, à informática, entre outros. As redes domóticas são polivalentes, ou seja, permitem realizar diferentes funções a fim de simplificar a complexidade da instalação da rede. A mesma rede domótica assegura, por exemplo: as funções de segurança, conforto e gestão de recursos, entre outros. A rede pode estar constituída de um ou vários padrões de comunicação de acordo com as funções que esse sistema domótico realiza.

3.2 Domótica Inteligente

Pode-se citar como características fundamentais num sistema inteligente: ter memória; ter noção temporal; fácil interação com os habitantes; capacidade de integrar todos os sistemas do ambiente; atuar em várias condições; facilidade de reprogramação e capacidade de auto-correção. Um sistema de domótica inteligente é a priori um sistema domótico com as características de um sistema inteligente.

O tema domótica inteligente é recente, por este motivo o seu conceito se confunde com o conceito de automação inteligente, ou melhor, processos de automação, os quais utilizam algumas técnicas de inteligência artificial para automaticamente tomar alguma decisão. Existem trabalhos que focam este tema centrado no conceito de automação inteligente.

Defini-se o conceito de Domótica Inteligente como a domótica onde se adapta as regras de automação do ambiente ao comportamento dos habitantes (TONIDANDEL; TAKIUCHI; MELO, 2004).

Um trabalho que foca a automação inteligente é (BOLZANI, 2004a), no qual é apresentado um simulador que controla dispositivos residenciais inteligentes; sendo considerados como inteligentes os dispositivos que tem autonomia para desenvolver uma tarefa básica e trocar informações com outros dispositivos. Neste trabalho o foco está no controle de atuadores segundo eventos detectados por sensores ou pela presença e ação de usuários, sendo aplicadas ações previamente configuradas aos eventos que acontecem.

Em (SIERRA et al, 2005) é feito o uso de técnicas de indução de conhecimento para inferir o conhecimento de especialistas em automatização de edifícios e assim gerar regras de automatização. As regras são posteriormente utilizadas em um sistema simulador para criar um edifício energeticamente eficiente.

O primeiro artigo relevante relacionado realmente ao tema Domótica Inteligente é *The Intelligent Room Project* (BROOKS, 1997), do MIT *Artificial Intelligence Lab*, o qual descreve uma sala inteligente fazendo uso de robótica, reconhecimento de voz e visão computacional que auxilia os habitantes na realização de certas tarefas. O projeto tem foco principal na interface entre o homem e a máquina.

Outro artigo importante é *Ada: Constructing a Synthetic Organism* (ENG et al, 2002), onde é apresentado um sistema que trata o ambiente como um organismo artificial, sendo possível modificar dinamicamente este ambiente através de diálogos com os habitantes para torná-lo mais funcional.

Um trabalho interessante é o *Adaptive Building Automation* (RUTISHAUSER; SCHÄFER, 2002a), posteriormente complementado em *Adaptive Building Intelligence* (RUTISHAUSER; SCHÄFER, 2002b), os quais apresentam um modelo de sistema domótico inteligente baseado em lógica Fuzzy e multi-agentes. Os trabalhos focam um sistema adaptativo ao ambiente e ao comportamento dos habitantes. A ênfase é o desenvolvimento de regras Fuzzy que podem ser dinamicamente alteradas de acordo com as mudanças no ambiente, através de uma política de punição e recompensa. O sistema foi testado em ambiente real e obteve sucesso. Apesar de ser um grande avanço no tema, observa-se que o sistema interage com os ambientes e habitantes de uma maneira geral, ou seja, não avalia eventos específicos. Não existe no sistema proposto uma etapa que valide as modificações feitas antes de repassá-las ao ambiente; também não foi apresentada uma lógica que detecte seqüências causais de eventos no tempo e a possibilidade de criar novas regras.

A Domótica inteligente não é simplesmente prover a uma residência um sistema dotado de controle central que possa aperfeiçoar certas funções inerentes à operação e administração da mesma.

Pode-se imaginar que uma residência inteligente é algo como uma residência interativa, dinâmica, ou seja, os sistemas de Domótica inteligente devem ter as características de um sistema inteligente e devem interagir com os habitantes da residência, aprendendo dinamicamente com seus comportamentos. Este aprendizado é permanente, pois os habitantes estão sempre mudando.

A inversão de responsabilidade no gerenciamento de uma residência deve ser o foco da inteligência da mesma. Os habitantes não devem se adaptar ao funcionamento pré-programado da residência, tendo de mudar seus métodos e gostos, mas sim a residência é quem deve se adaptar ao comportamento dos habitantes.

Importante notar que ao atuar segundo o comportamento dos habitantes, o sistema também permitirá a diminuição da intervenção do usuário na programação do mesmo, permitindo que usuários com menor familiarização ao tema possam utilizar o sistema.

3.3 Sistema ABC

Um sistema anteriormente apresentado (TONIDANDEL; TAKIUCHI; MELO, 2004) que atua segundo o comportamento dos habitantes é o sistema ABC (Automação Baseada em Comportamento).

O processo normal de criação de regras é aquele onde o habitante é quem cria as regras, inserindo-as em um sistema. O sistema ABC foi testado e demonstrou através de simulações que é possível reverter o processo normal de criação de regras.

O sistema concentra-se no algoritmo de aprendizado ID3 (QUINLAN, 1986), o qual é responsável pelo processo de adaptação do sistema de automação.

A arquitetura do sistema ABC (fig. 3.1) define a existência de sensores (detectores de presença, medidores de temperatura, medidores de luminosidade, entre outros.), atuadores (interruptores de luz, ar-condicionado, entre outros.), bancos de dados e demais elementos necessários para criação e controle das regras. Os sensores1 agem diretamente com os atuadores e podem sofrer interferência humana em seu estado, os sensores2 agem sem a interferência humana. O controlador monitora constantemente os sensores de cada ambiente, verificando se existe alguma regra de automação armazenada no seu Banco de Dados ou nas Regras de Segurança que se aplique ao estado dos sensores.

Para cada atuador da casa existe um banco de aquisição de dados de comportamento, o qual armazena informações sobre mudanças ocorridas no atuador; este Banco de Dados é alimentado com os eventos que ocorrem no atuador e os respectivos dados dos sensores vinculados ao atuador. Como exemplo, pode-se ter o atuador “Ar-condicionado” e os sensores “Temperatura”, “Luminosidade”, “Horário” e “Presença”. Quando o estado do Ar-condicionado muda, por ação do habitante, os dados do próprio atuador e mais os dados dos sensores são armazenados em uma posição de memória do Banco de Dados de Aquisição de Comportamento (Ar-condicionado=Ligado; Temperatura=Alta; Luminosidade=Alta; Horário=Noite; Presença=Sim).

Em dado momento, com certo número de eventos armazenados, o Banco de Dados é inserido no algoritmo de aprendizado, o qual generaliza os dados e cria regras. As regras são armazenadas no Banco de Dados de Regras Ativas. Existe também outro Banco de Dados onde estão as Regras de Segurança. No exemplo citado, uma regra aprendida poderia ser: SE Temperatura=Alta E Horário=Noite E Luminosidade=Alta E Presença=Sim ENTÃO Ar-condicionado=Ligado. Assim, quando os sensores indicarem Temperatura=Alta, Horário=Noite, Luminosidade=Alta e Presença=Sim o ar-condicionado será automaticamente ligado pelo Sistema ABC. Ou seja, a partir do momento em que novos eventos acontecem, de acordo com as ações do habitante, é feita uma varredura no Banco de Dados de Regras Ativas para avaliar se alguma regra deve ser aplicada e realizar uma ação no atuador.

A manutenção das regras é simples, quando uma regra fica um determinado tempo sem ser utilizada, ela é removida. A base de dados armazena a data da última utilização da

regra e caso a regra não seja acionada por um dado período de tempo, a mesma é eliminada. Também é possível remover regras manualmente pelo habitante, caso alguma regra criada não agrade o mesmo (TONIDANDEL; TAKIUCHI; MELO, 2004).

Dada a simplicidade das regras a serem criadas e a maneira como podem ser expressas as informações dos elementos utilizados no sistema, a lógica proposicional é suficiente para ser utilizada na criação das regras, não sendo necessário atuar com uma lógica mais complexa, como, por exemplo, a lógica de primeira ordem.

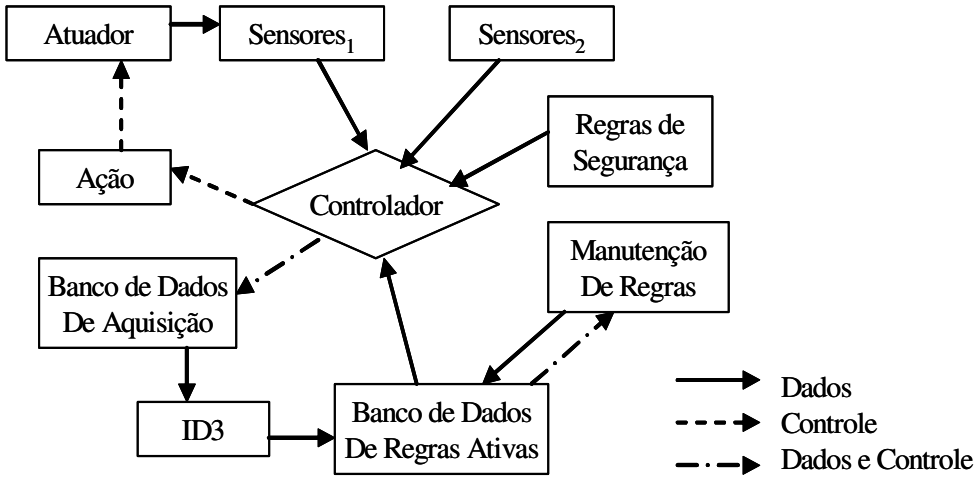


Figura 3.1 – Arquitetura do Sistema ABC, adaptada de (TONIDANDEL; TAKIUCHI; MELO, 2004).

Apesar de muito interessante, o sistema possui deficiências. Ele não detecta seqüências causais de eventos no tempo, se um evento de atuador acontece depois de pouco ou muito tempo de um evento de sensor isto não é considerado. Neste sistema, com o uso do ID3, é possível trabalhar somente com variáveis lógicas, não é possível trabalhar com valores contínuos. Outra deficiência é o fato de regras criadas pelo ID3 se tornarem diretamente Regras Ativas, isto pode desagradar o habitante da casa.

4 SISTEMA ABC+ PROPOSTO

O sistema ABC, em sua versão inicial, marcou a implementação de regras em automação residencial através da observação do comportamento de um habitante em uma casa, entretanto é possível observar que ele possui limitações que podem ser aprimoradas.

Com o intuito de corrigir o sistema ABC e diminuir suas limitações, propõem-se o sistema ABC+, o qual possui arquitetura (fig.4.1) parecida com o sistema inicial, porém seu funcionamento apresentado no fluxograma (fig. 4.6) mais adiante é bastante diferente.

As principais diferenças são a janela de observação de eventos, as Regras Embrionárias, o uso do C4.5 (QUINLAN, 1993) e o novo processo de desenvolvimento e manutenção das regras. Estas diferenças serão detalhadas a partir da seção 4.1.

O algoritmo C4.5 foi utilizado em função de ser um algoritmo bastante conhecido e muito utilizado na área de árvores de indução de regras. Ele permite trabalhar com valores contínuos para os atributos e também com valores de atributos desconhecidos. Outros algoritmos poderiam ter sido escolhidos, porém não é foco do trabalho eleger o melhor algoritmo para a criação de regras, mas sim utilizar um que seja conceituado para a sua função.

A arquitetura do sistema ABC+ (fig. 4.1) define um habitante interagindo com sensores e atuador. Existem ainda na arquitetura, três bancos de dados (Eventos, Regras Ativas, onde estão também as de segurança da residência, e Regras Embrionárias), a janela de observação, o C4.5, a lógica de manutenção de regras e o controle central, o qual possui a lógica central do sistema. Para cada atuador da casa deve existir uma estrutura paralela de bancos de dados, ou seja, cada atuador possui três bancos de dados. Os demais elementos da arquitetura podem ser compartilhados.

Quando algum sensor ou o atuador muda de estado, seja pela ação do habitante ou pela natureza, um evento é gerado. Se o evento for do atuador, a lógica da janela de observação irá filtrar o evento e o mesmo poderá ser ou não armazenado no Banco de Dados de eventos. Em dado momento, com um número pré-configurado de eventos armazenados, o Banco de Dados de eventos é inserido no algoritmo de aprendizado, o qual generaliza os dados e cria regras. As regras são armazenadas no Banco de Dados de Regras Embrionárias e após uma etapa de validação as mesmas podem ascender ao Banco de Dados de Regras Ativas, o qual possui também as Regras de Segurança da residência; estas últimas somente são modificadas se manipuladas diretamente no sistema. A lógica de manutenção de regras atua para estabelecer quais regras ficam em quais bancos, isto será detalhado mais a diante.

Quando acontecem novos eventos nos sensores é feita uma varredura no Banco de Dados de Regras Ativas para avaliar se alguma regra deve ser aplicada e realizar uma ação no atuador.

Um dormitório de uma casa onde existem um sensor de entrada no ambiente, um sensor de saída do ambiente e um atuador para ligar e desligar uma lâmpada servirá de exemplo para detalhar as implementações feitas no sistema ABC+.

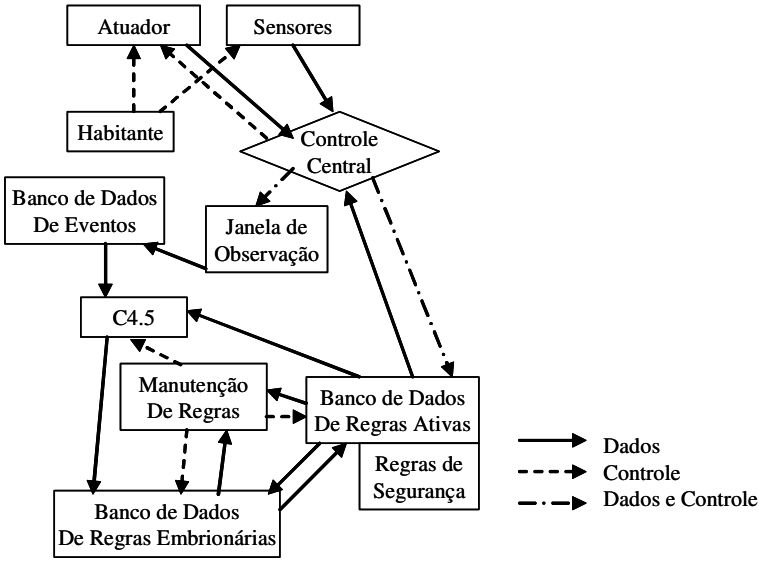


Figura 4.1 – Arquitetura do Sistema ABC+.

4.1 A Janela de Observação

Nas condições em que o habitante entra no dormitório e imediatamente acende a lâmpada ou quando sai do dormitório e imediatamente desliga a lâmpada, pode-se facilmente assimilar regras, que seriam: SE habitante entra ENTÃO acenda lâmpada; SE habitante sai ENTÃO apague lâmpada.

Note que, se o habitante entra no dormitório e após uma hora ele acende a lâmpada, os sensores detectarão os mesmos dados da situação em que ele imediatamente acende a lâmpada após entrar no dormitório. Isto leva a criação de uma regra errada.

Para eliminar esta limitação a janela de observação é utilizada. A janela de observação consiste em se armazenar e comparar cada evento anterior e posterior ao evento em análise, inclusive com horários (fig. 4.2). Os eventos em análise são eventos de atuador, ou seja, eventos em que o atuador muda de estado, no exemplo a lâmpada. Eventos de sensores não são utilizados para gerar regras, mas sim para ativarem uma regra, se a mesma existir. Na

janela de observação os eventos anterior e posterior podem ser eventos de sensores, pois servirão de comparação e não para criação de regra.

A janela de observação trata cada evento de atuador isoladamente, juntamente com seus eventos de sensor anterior e posterior. O objetivo é identificar uma causa (evento no sensor) e um efeito (evento no atuador). Se não existe uma causa e um efeito próximos no tempo, não faz sentido armazenar tal efeito para que o mesmo gere futura regra. Uma casualidade não deve gerar uma regra, somente eventos repetitivos (padrões ou rotinas) devem gerar regras. Um evento de atuador não tem vínculo com o próximo evento de atuador.

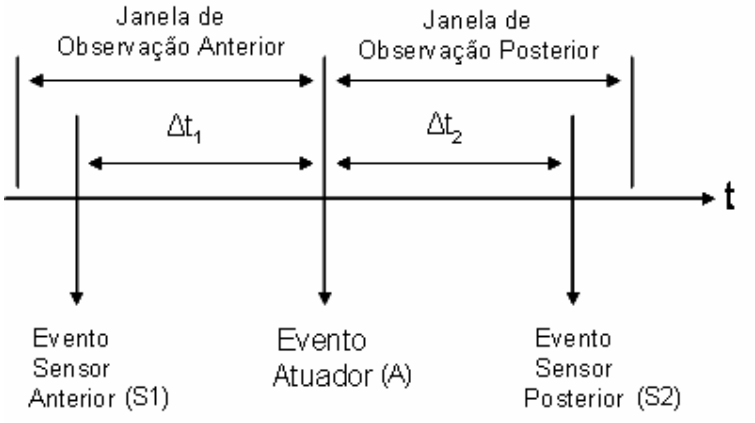


Figura 4.2 – Janela de Observação no Sistema ABC+.

Com um exemplo é possível entender a lógica da janela de observação (fig. 4.3). Quando o habitante entra no ambiente ele dispara um evento de sensor (S1), que é armazenado temporariamente com seu devido horário. Após algum tempo ele acende a lâmpada. Este evento de atuador (A) é armazenado e comparado com o evento anterior (S1), se o anterior aconteceu dentro de um determinado intervalo de tempo (Δt_1) menor do que o valor da janela de observação então significa que o evento do atuador está vinculado ao evento anterior (Δt_1 < janela de observação Anterior), neste caso o evento do atuador é armazenado para gerar uma futura regra.

Quando existe vínculo com o evento anterior não se realiza a comparação com o evento posterior (S2) e o sistema volta à situação de início. Porém, se foi constatado que não existe vínculo entre evento do atuador e o evento anterior, é necessário verificar o evento posterior. O horário do evento atuador é armazenado e aguarda-se a ocorrência de um novo evento. Quando o evento (S2) acontece se compara a diferença de horários entre ele e o evento atuador (Δt_2). Se o evento posterior aconteceu dentro do determinado intervalo de

tempo referente à janela de observação, então significa que o evento do atuador está vinculado ao evento posterior (Δt_2 < janela de observação Posterior), caso contrário não existe vínculo.

No caso de estarem vinculados armazenam-se os dados do evento de sensor posterior para gerar futura regra, pois é este evento que reflete as reais condições para o atuador ter sido mudado.

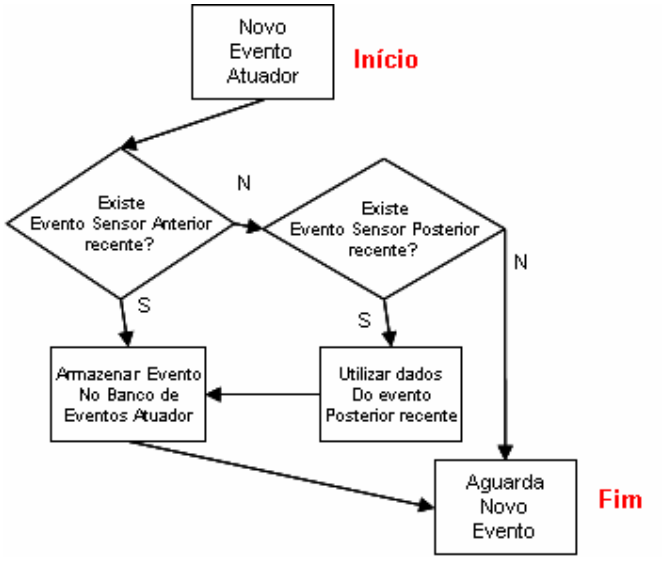


Figura 4.3 – Fluxograma da lógica da Janela de Observação.

O limitante da janela de observação são os períodos de tempo anterior e posterior. Se a causa (evento de sensor) e efeito (evento de atuador) não estiverem dentro destes limites, o evento é considerado uma casualidade e descartado.

4.2 As Regras Embrionárias

No sistema ABC, sempre que o ID3 cria uma regra, ela é imediatamente colocada como Regra Ativa. Sendo assim, não existe uma etapa de validação da regra. A primeira vez que o habitante gerar as condições dos sensores iguais às condições da regra, a mesma será executada e alguma ação no atuador será feita. Se o habitante contraria a regra nada acontece, e se ele quiser acabar com a regra terá de removê-la manualmente.

Ao criar uma regra, deve existir uma etapa de validação da mesma, ou seja, ter certeza que a regra é aceita pelo habitante. No sistema ABC+, toda vez que novas regras são criadas, elas são armazenadas inicialmente no banco de Regras Embrionárias.

As Regras Embrionárias são regras que estão em validação. Quando o habitante gerar condições dos sensores iguais às condições de uma Regra Embrionária, o que acontece é que

a regra ganha pontos positivos. Somente quando a regra atinge certa pontuação superior é que ela passa para o banco de Regras Ativas.

Por outro lado, se o habitante contrariar a regra, ela ganha pontos negativos e ao atingir uma pontuação inferior a mesma é eliminada do banco de Regras Embrionárias (fig.4.4). Na figura 4.4, "outras ações" indicam partes do fluxograma que não estão detalhadas na mesma, ou seja, "outras ações" representam um bloco macro com lógicas que realizam outras ações presentes no sistema proposto.

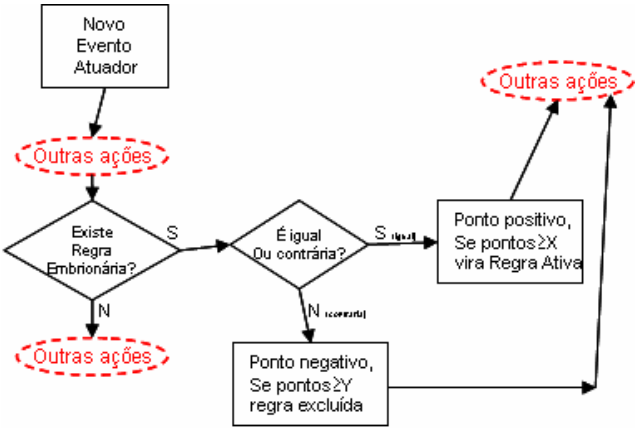


Figura 4.4 – Fluxograma da lógica de validação da Regra Embrionária.

4.3 O Desenvolvimento e Manutenção das Regras

A maneira como se mantêm as regras em uma casa influencia a interação do sistema com o habitante. A inserção e remoção de regras têm de ser o mais sutil possível, caso contrário trará desconforto ao usuário e podem incorrer em desestabilização das regras.

No sistema ABC+ existe dois tipos de regras, as Ativas e as Embrionárias. A figura 4.5 apresenta as relações entre as regras. No decorrer deste capítulo as relações serão detalhadas.

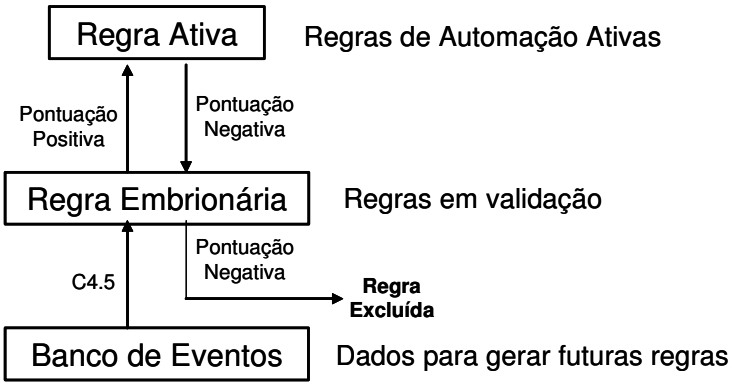


Figura 4.5 – Fluxograma da lógica de validação da Regra Embrionária.

Para o desenvolvimento e manutenção de regras no sistema ABC+ são utilizados três bancos de dados para cada atuador da casa (tab. 4.1, tab.4.2 e tab. 4.3).

A tabela de Eventos por Atuador (A1, A2, ...An) tem os seguintes campos:

Tabela Eventos = {N° Evento, Sensor1(S1), Sensor2(S2), ..., SensorN(Sn), Atuador An}.

A tabela de Regras Ativas por Atuador (A1, A2, ...An) tem os seguintes campos:

Tabela Regras Ativas = {N° Evento, Sensor1(S1), Sensor2(S2), ..., SensorN(Sn), Atuador An, ATIV, EXC}.

A tabela de Regras Embrionárias por Atuador (A1, A2, ...An) tem os seguintes campos:

Tabela Regras Embrionárias = {N° Evento, Sensor1(S1), Sensor2(S2), ..., SensorN(Sn), Atuador An, OK, NOK}.

Tabela 4.1 – Eventos por Atuador (A1, A2, ...An).

Evento	Sensor 1 (S1)	Sensor 2 (S2)	.....	Sensor N (Sn)	Atuador An
1	Valor A	Valor A	.....	Valor A	Valor A
2	Valor C	Valor B	.....	Valor A	Valor B
3	Valor B	Valor A	.....	Valor B	Valor A
4	Valor N	Valor N	.....	Valor N	Valor N
....	....	....	....	....	....

Tabela 4.2 – Regras Ativas por Atuador (A1, A2, ...An).

Regra	Sensor 1 (S1)	Sensor 2 (S2)	.....	Sensor N (Sn)	Atuador An	ATIV	EXC
1	Valor A	Valor A	.....	Valor A	Valor A	6	0
2	Valor C	Valor B	.....	Valor A	Valor B	4	0
3	Valor B	Valor A	.....	Valor B	Valor A	3	2
4	Valor N	Valor N	.....	Valor N	Valor N	1	1
....	....	....	....	....	....	....	....

Tabela 4.3 – Regras Embrionárias por Atuador (A1, A2, ...An).

Regra	Sensor 1 (S1)	Sensor 2 (S2)	.....	Sensor N (Sn)	Atuador An	OK	NOK
1	Valor A	Valor A	.....	Valor A	Valor A	2	1
2	Valor C	Valor B	.....	Valor A	Valor B	5	2
3	Valor B	Valor A	.....	Valor B	Valor A	1	0
4	Valor N	Valor N	.....	Valor N	Valor N	6	0
....	....	....	....	....	....	....	....

Serão utilizadas as seguintes definições:

- BDEventos – Banco de Dados de Eventos por Atuador.
- BDAtivas – Banco de Dados de Regras Ativas.
- ATIV – Campo do BDAtivas utilizado para pontuar positivamente uma regra.
- EXC - Campo do BDAtivas utilizado para pontuar negativamente uma regra.
- BDEmbrio – Banco de Dados de Regras Embrionárias.
- OK - Campo do BDEmbrio utilizado para validar uma regra.
- NOK - Campo do BDEmbrio utilizado para excluir uma regra.

Os novos eventos que surgem nos atuadores da casa, e passam pelo crivo da janela de observação, são armazenados no Banco de Dados de Eventos por Atuador (tab. 4.1).

Na tabela 4.1 cada linha representa um evento de atuador armazenado. A informação que consta na linha é o valor do atributo atuador e os valores dos atributos dos sensores para um determinado evento. Um exemplo de evento e atributos pode ser: atributo atuador igual a “Lâmpada” e seus valores “Acesa” e “Apagada”, atributo sensor 1 igual a “Temperatura” e seus valores “Alta”, “Normal” e “Baixa”, atributo sensor 2 igual a “Luminosidade” e seus valores “Alta”, “Normal” e “Baixa”, atributo sensor 3 igual a “Umidade” e seus valores “Alta”, “Média” e “Baixa” e atributo sensor 4 igual a “Porta” e seus valores “Entrada” e “Saída”. Um evento pode ser:

Tabela 4.4 – Exemplo de evento armazenado no BDEventos.

Evento	Temperatura	Luminosidade	Umidade	Porta	Lâmpada
1	Alta	Alta	Média	Entrada	Acesa

Quando o Banco de Dados de eventos de atuador atinge um número pré-determinado de eventos ocorre a sua inserção no algoritmo C4.5 para a geração de novas regras.

Estas novas regras são inicialmente comparadas com as Regras Ativas e Regras Embrionárias, para que as regras repetidas sejam eliminadas.

As novas regras geradas, que não forem repetidas, vão para o banco de Regras Embrionárias (tab. 4.3). No Banco de Regras Embrionárias existem para cada regra, além dos campos de sensores e atuador, os campos OK e NOK, os quais servem para o controle da validação da regra. Quando uma Regra Embrionária é gerada ela possui os campos OK e NOK iguais a zero. Seguindo o descrito anteriormente, uma possível regra no BDEmbrio é:

Tabela 4.5 – Exemplo de Regra Embrionária no BDEmbrio.

Regra	Temperatura	Luminosidade	Umidade	Porta	Lâmpada	OK	NOK
1	Baixa	Média	Alta	Saída	Apagada	2	1

Sempre que ocorre um novo evento de atuador este é comparado primeiro com as Regras Ativas, para verificar se alguma Regra Ativa está sendo contrariada ou não, depois este evento é comparado com as Regras Embrionárias. Caso exista Regra Embrionária relacionada ao evento é necessário dar pontos à mesma. Se o evento confirmou a regra, ao valor do campo OK é somado mais um, caso o evento contrarie a regra o valor do campo NOK recebe mais um. Quando o campo OK atinge certo valor, a Regra Embrionária é transformada em Regra Ativa. Quando o campo NOK atinge certo valor, a Regra Embrionária é removida do banco de regras.

Os campos OK e NOK trabalham como porcentagens de acertos e erros. Por exemplo, se o valor no campo OK para a regra ser validada é oito e o valor no campo NOK para a regra ser excluída é dois, significa que quando temos 80% ou mais de acertos a regra é validada ou quando temos 20% ou mais de erros a regra é excluída.

O Banco de Regras Ativas (tab. 4.2) também possui uma lógica de manutenção. É neste banco que estão as Regras de Segurança (são regras para a segurança da casa, como por exemplo, se existe fogo então cortar energia e gás), as quais não sofrem modificações, e as regras a serem executadas na casa. Sempre que algum evento de sensores acontece o banco de Regras Ativas é consultado, se existir alguma regra relacionada ao evento a mesma é aplicada, assim uma ação é feita no atuador. Neste caso soma-se ao valor do campo ATIV o valor um.

O banco de Regras Ativas possui uma quantidade limitada de regras. As regras estão ordenadas de acordo com o valor do campo ATIV. Quando uma nova regra é inserida no BDAtivas e este está com sua capacidade máxima, então é necessário retirar a regra com menor pontuação ATIV e colocar esta regra de volta ao banco de Regras Embrionárias.

Tabela 4.6 – Exemplos de Regras Ativas no BDAtivas.

Regra	Temperatura	Luminosidade	Umidade	Porta	Lâmpada	ATIV	EXC
1	Baixa	Alta	Alta	Saída	Apagada	5	1
2	Média	Média	Baixa	Entrada	Acesa	3	0

Um evento de atuador também recebe atenção por parte do BDAtivas. Caso o evento atuador seja contrário a alguma regra existente no BDAtivas, soma-se ao valor do campo EXC desta regra em questão o valor um. Se o campo EXC ultrapassar um valor pré-determinado, esta regra é excluída do BDAtivas.

No momento de criação de novas regras (atuação do C4.5) também é checado se alguma regra do BDAtivas ou do BDEmbrio ficou velha. Para isto subtrai-se o valor um do campo ATIV de todas as Regras Ativas. As Regras Ativas que neste momento tiverem seu campo ATIV abaixo de certo valor serão rebaixadas ao BDEmbrio. As Regras Embrionárias que neste momento tiverem seu campo OK abaixo de certo valor serão excluídas.

Importante ressaltar que o sistema aprende o que é repetitivo. Eventos não repetitivos não irão validar as Regras Embrionárias para as mesmas se tornarem Regras Ativas.

O Sistema ABC+ trabalha com somente um habitante na residência. Não existem vínculos entre atuadores e entre atuadores e sensores. As saídas dos atuadores devem ter seus valores representados de maneira discreta. Estas limitações apontadas devem ser tratadas em trabalhos futuros.

Para que o sistema possa ser estendido para aplicações reais e mais complexas será ainda necessário identificar como lidar com os possíveis *loops* nas regras criadas.

Um *loop* pode fazer com que uma regra ao ser ativada leve à ativação direta de uma outra regra ou ativação indireta de regra por mudança de algum sensor.

Na manutenção de regras foram citadas algumas vezes que os campos OK, NOK, ATIV, EXC e a quantidade de eventos no C4.5 ao atingirem um valor pré-determinado fazem com que regras sejam criadas, promovidas ou eliminadas. Estes valores pré-determinados, na verdade podem ser configurados através de variáveis.

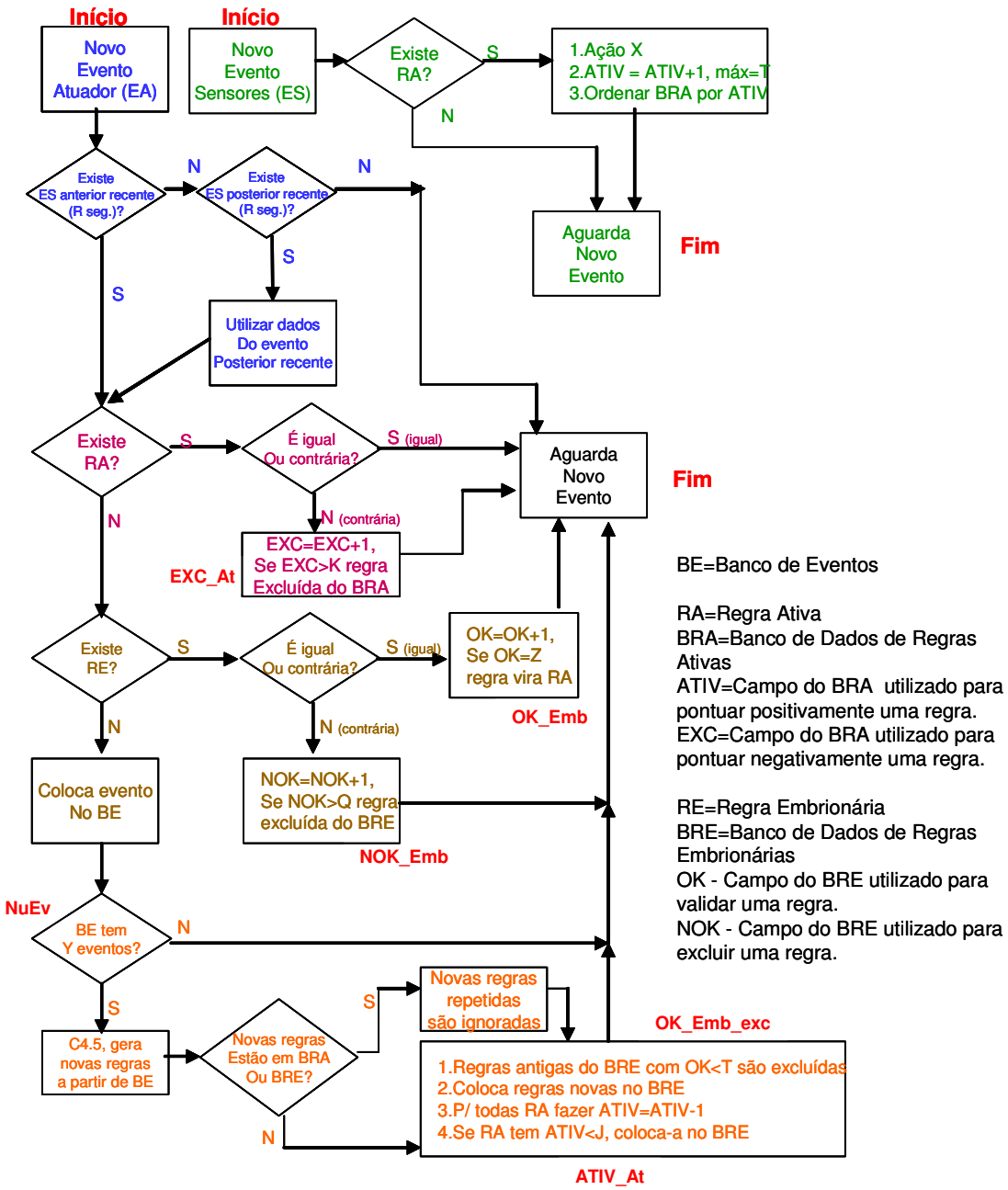


Figura 4.6 – Fluxograma do Funcionamento do Sistema ABC+.

Torna-se necessário definir os nomes destas variáveis do sistema, as quais serão objeto de avaliação neste trabalho:

- NuEv = Número de eventos no BDEventos para acionar o C4.5;
- OK_Emb = Valor do campo OK do BDEmbrio para regra virar Regra Ativa;
- NOK_Emb = Valor do campo NOK do BDEmbrio para regra ser excluída;
- EXC_At = Valor do campo EXC do BDAtivas para excluir a regra;

- ATIV_At = Valor do campo ATIV do BDAtivas para regra ir para o BDEmbrio, devido a desuso;
- OK_Emb_exc = Valor do campo OK do BDEmbrio para a regra ser excluída por desuso.

O fluxograma do funcionamento do sistema ABC+ (fig. 4.6) reflete as informações descritas nesta seção.

```
Evento de Sensor
(BDAtivas: conjunto de Regras Ativas de um atuador,
ATIV: Campo do BDAtivas utilizado para pontuar positivamente uma regra);

Início
Se existe Regra Ativa relacionada ao evento,
             Aplicar a regra, fazer campo ATIV da regra igual a ATIV+1,
             ordenar BDAtivas por valor de ATIV;
Fim
```

Figura 4.7 – Algoritmo Evento de Sensor.

```
Evento de Atuador
(BDEventos: conjunto de Eventos por Atuador,
BDAtivas: conjunto de Regras Ativas de um atuador,
BDEmbrio: conjunto de Regras Embrionárias de um atuador,
ATIV: Campo do BDAtivas utilizado para pontuar positivamente uma regra,
EXC: Campo do BDAtivas utilizado para pontuar negativamente uma regra,
OK: Campo do BDEmbrio utilizado para validar uma regra,
NOK: Campo do BDEmbrio utilizado para excluir uma regra,
NuEv: Número de eventos no BDEventos para acionar o C4.5,
OK_Emb: Valor do campo OK do BDEmbrio para regra virar Regra Ativa,
NOK_Emb: Valor do campo NOK do BDEmbrio para regra ser excluída,
EXC_At: Valor do campo EXC do BDAtivas para excluir a regra,
ATIV_At: Valor do campo ATIV do BDAtivas para regra ir para o BDEmbrio, devido a desuso,
OK_Emb_exc: Valor do campo OK do BDEmbrio para a regra ser excluída por desuso);

Início
Se existe Regra Ativa igual ao evento,
             Vá para o Fim;
Se existe Regra Ativa contrária ao evento,
             Fazer campo EXC da regra igual a EXC+1;
             SE EXC maior que EXC_At,
                    Rebaixar Regra Ativa a Regra Embrionária;
Se existe Regra Embrionária igual ao evento,
             Fazer campo OK da regra igual a OK+1;
             SE OK igual a OK_Emb,
                    Promover a regra a Regra Ativa;
Se existe Regra Embrionária contrária ao evento,
             Fazer campo NOK da regra igual a NOK+1;
             SE NOK maior que NOK_Emb,
                    Excluir a Regra Embrionária;
Senão
             Colocar o evento no BDEventos;
             Se BDEventos igual a NuEv,
                    Acionar C4.5 para gerar novas regras;
                    Se novas regras já estão no BDAtivas ou BDEmbrio,
                           Ignorar as regras repetidas;
                    Se alguma Regra Embrionária tem campo OK menor que OK_Emb_exc,
                           Excluir Regra Embrionária;
                    Colocar regras novas no BDEmbrio;
                    Fazer campo ATIV das Regras Ativas igual a ATIV-1;
                    SE alguma Regra Ativa tem campo ATIV menor do que ATIV_At,
                           Rebaixar Regra Ativa a Regra Embrionária;
Fim
```

Figura 4.8 – Algoritmo Evento de Atuador.

O desenvolvimento e manutenção das regras podem ser representados também através de seus algoritmos. São apresentados nas figuras 4.7 e 4.8 os algoritmos para quando ocorre um Evento de Sensor e um Evento de Atuador.

Outra maneira de representar o funcionamento do sistema ABC+ é a Rede de Petri presente no Apêndice 1.

O algoritmo da figura 4.7 não engloba a janela de observação de eventos, a qual já foi detalhada no item 4.1.

4.4 Inconsistência nas Regras

Um ponto importante que deve ser abordado é a inconsistências nas regras criadas. No sistema ABC+ não existe inconsistência nas regras.

A inconsistência implica em ter regras que são contrárias entre si ou ter combinações de sensores que ativam mais do que uma regra.

Quando um evento de atuador é armazenado no Banco de Eventos do Atuador ele futuramente irá gerar uma Regra Embrionária. Esta Regra Embrionária será pontuada positivamente até se transformar em uma Regra Ativa. A cada vez que uma combinação de sensor (evento de sensor) atender a esta Regra Ativa, a mesma é executada.

Se o habitante passa a ter um comportamento contrário à Regra Ativa, fazendo com que o atuador tenha um valor diferente do valor de atuador da regra, o que acontece é que a regra receberá pontos no campo EXC desta Regra Ativa, até o momento que a regra é rebaixada a Regra Embrionária. Caso o habitante ainda continue repetindo o evento no atuador, a Regra Embrionária receberá pontos no campo NOK, até que a mesma é excluída.

O habitante mantém a nova rotina, gerando novos eventos no atuador. Estes eventos agora não contrariam a regra, pois a regra já foi excluída. Os eventos são então armazenados no Banco de Eventos do Atuador para gerarem nova Regra Embrionária e posterior Regra Ativa.

Pode-se observar que não existirão inconsistências, pois sempre ocorrerá primeiro a exclusão de regras existentes antes da criação de novas regras que as possam contrariar.

A análise de inconsistências nas regras finalizam este capítulo sobre o Sistema ABC+, no próximo capítulo serão apresentadas as simulações realizadas no trabalho.

5 SIMULAÇÃO DO SISTEMA ABC+

Para realização dos testes deste trabalho foram criados dois simuladores envolvendo o sistema proposto. Um simulador foi desenvolvido para avaliar a lógica do sistema e o outro simulador foi feito como objetivo de manipulação das variáveis do sistema, para analisar em detalhe os comportamentos destas variáveis.

5.1 Simulação da Lógica do Sistema

Um simulador foi desenvolvido para confirmar o funcionamento da lógica do sistema ABC+. O ambiente simulado possui um cômodo (dormitório) com um atuador (lâmpada), os seguintes sensores e seus respectivos valores possíveis:

- Temperatura: Alta, Normal, Baixa;
- Luminosidade: Alta, Normal, Baixa;
- Umidade: Alta, Média, Baixa ;
- Porta: Entrada, Saída;

O Atuador possui os valores possíveis Acesa e Apagada.

Para a implementação do simulador foi utilizado o software Delphi. O algoritmo C4.5 utilizado é o original (QUINLAN, 1993), porém codificado para o sistema operacional Windows.

Os sensores podem ter seu estado alterado através de botões, conforme a figura 5.1. As simulações foram feitas manipulando estes botões e assim gerando eventos.

No trabalho desenvolvido, os simuladores são ferramentas para verificação do que foi proposto, não será dada ênfase ao desenvolvimento dos mesmos e sim ao sistema proposto, portanto não será feito detalhamento do código fonte dos simuladores.

Um exemplo de evento é apertar o valor Normal, do sensor Temperatura. Isto gera um evento de sensor que fica armazenado temporariamente, inclusive com horário, e pode ser visualizado no quadro um do simulador. Este evento pode: levar à execução de uma das Regras Ativas, as quais podem ser observadas no quadro quatro do simulador ou atuar na validação ou rejeição de uma das Regras Embrionárias, as quais podem ser vistas no quadro três do simulador ou servir de referência na janela de observação.

Outro exemplo é apertar o valor Apagada do atuador Lâmpada. Isto gera um evento de atuador, o qual passa pela lógica da janela de observação e se for validado é armazenado no

Banco de Dados de eventos do atuador. No quadro dois do simulador é possível ver os eventos de atuador armazenados.

O sistema teve o funcionamento esperado, conforme descrito no capitulo 4, criando regras novas e aplicando as mesmas em eventos posteriores.

A lógica da janela de observação funcionou perfeitamente, ou seja, os eventos indesejados foram filtrados, permitindo que somente os eventos de atuador que possuem vínculos com um evento de sensor anterior ou posterior fossem para o Banco de Eventos.

Quando o BDEventos atingiu a quantidade de eventos configurada, os dados foram inseridos no algoritmo C4.5 e Regras Embrionárias foram criadas. As Regras Embrionárias atuaram como uma etapa de validação de regras, somente promovendo regras à Regras Ativas após as confirmações das mesmas. As Regras Embrionárias que foram contrariadas acabaram sendo eliminadas do BDEmbrio.

As Regras Ativas também funcionaram como especificado, foram mantidas as regras desejadas e eliminadas as regras que foram contrariadas ou que ficaram em desuso.

Figura 5.1 – Simulador para teste da lógica do sistema ABC+.

5.2 Simulador para Avaliação das Variáveis do Sistema

Com a implementação do primeiro simulador surgiu a necessidade de avaliar melhor o funcionamento do sistema, para isto foi necessário avaliar como se comportam as variáveis do sistema ABC+ quando submetido ao comportamento de um agente (habitante).

O primeiro ponto a ser desenvolvido antes da elaboração do novo simulador foi a identificação dos parâmetros de erro e desempenho que melhor permitem avaliar o sistema. Em outras palavras, foi necessário definir quais informações deveriam ser extraídas do sistema, as quais permitem comparar o funcionamento do sistema quando os valores das variáveis são manipulados.

Variáveis do sistema ABC+ que necessitavam ser avaliadas são:

- NuEv = Número de eventos no BDEventos para acionar o C4.5 [Y];
- OK_Emb = Valor do campo OK do BDEmbrio para regra virar Regra Ativa [Z];
- NOK_Emb = Valor do campo NOK do BDEmbrio para regra ser excluída [Q];
- EXC_At = Valor do campo EXC do BDAtivas para excluir a regra [K];
- ATIV_At = Valor do campo ATIV do BDAtivas para regra ir para o BDEmbrio, devido a desuso [J];
- OK_Emb_exc = Valor do campo OK do BDEmbrio para a regra ser excluída por desuso [W].

As variáveis listadas podem ter seus valores configurados no simulador (figura 5.2).

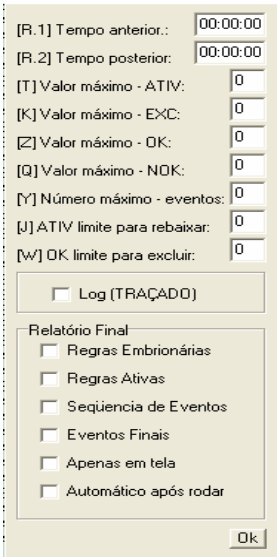


Figura 5.2 – Tela do simulador para configuração das variáveis do sistema ABC+.

O segundo simulador foi baseado no primeiro simulador desenvolvido, entretanto com as seguintes modificações:

Existem um Atuador e 6 sensores (Sensor1, Sensor2, Sensor3, Sensor4, Sensor5 e Sensor6). Para o atuador existem os estados A, B e C, para os sensores 1, 2, 3 e 4 existem os estados A, B e C, para o sensor 5 existem os estados A, B, C e D e para o sensor 6 existem os estados A, B, C, D e E.

Para as simulações foi criado um Banco de Dados de eventos para refletir as atividades de um usuário durante um período de tempo; estes eventos são inseridos no sistema, em seguida observa-se o comportamento das regras que são produzidas.

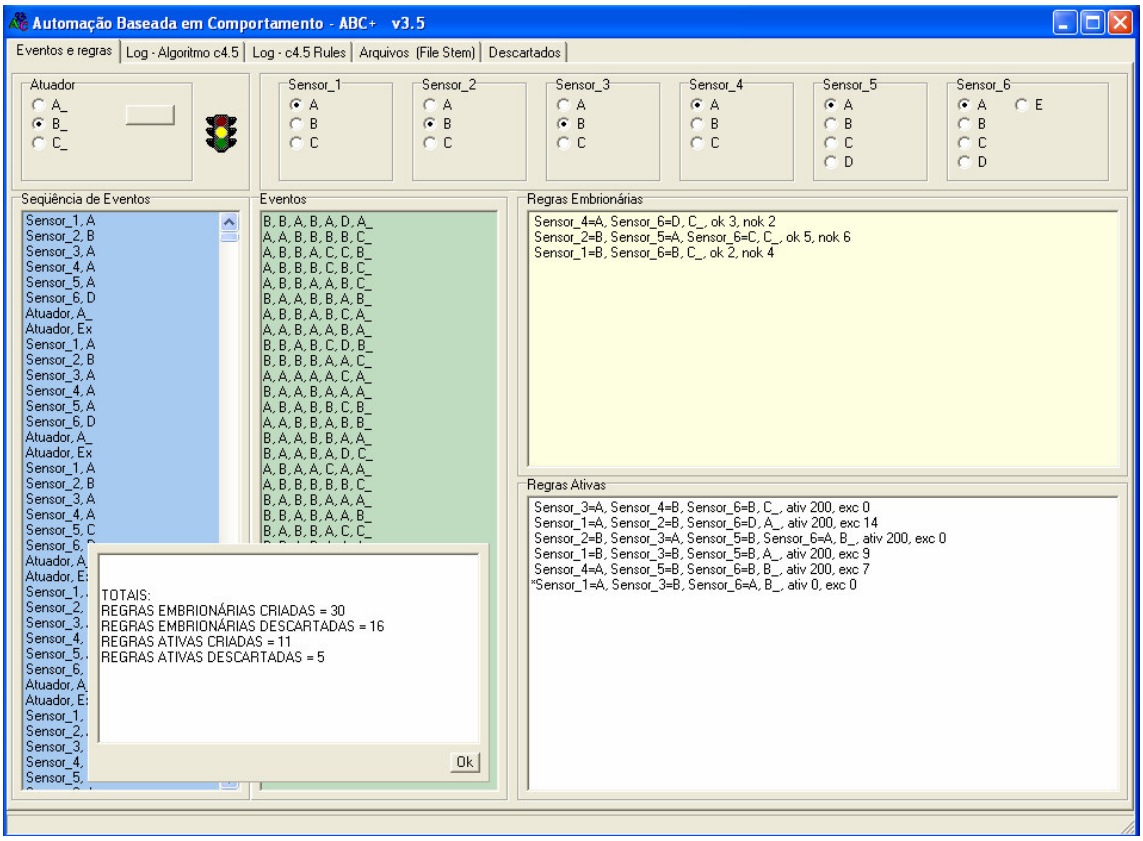


Figura 5.3 – Simulador para teste das variáveis do sistema ABC+ (Eventos e Regras).

Na simulação, o simulador busca o primeiro evento no Banco de Dados externo e executa sua programação, a seguir busca o segundo evento e executa sua programação; isto é feito sucessivamente até que o último evento seja buscado. Para as simulações não houve limitação do número de Regras Ativas a serem armazenadas no BDAtivas, tal limitação imporia mais uma variável no sistema, a qual influenciaria nas outras. A limitação do número máximo de regras do BDAtivas está diretamente relacionada com a percepção do habitante

sobre as regras, portanto deverá ser avaliada em testes com residências e habitantes reais, o que ficará para trabalhos futuros.

Como se pode observar nas figuras 5.3 e 5.4, o simulador permite visualizar quais eventos foram inseridos no BDEventos, quais Regras Embrionárias foram criadas e quais foram descartadas, quais Regras Ativas foram criadas e quais foram descartadas.

Ao final de cada simulação o simulador devolve um quadro (*pop-up*) com os valores dos parâmetros de desempenho. Outra informação que o simulador fornece são quais regras e em qual momento da simulação as mesmas foram criadas e eliminadas, permitindo uma análise qualitativa. O relatório produzido pelo simulador que permite a análise qualitativa será doravante chamado de traçado.

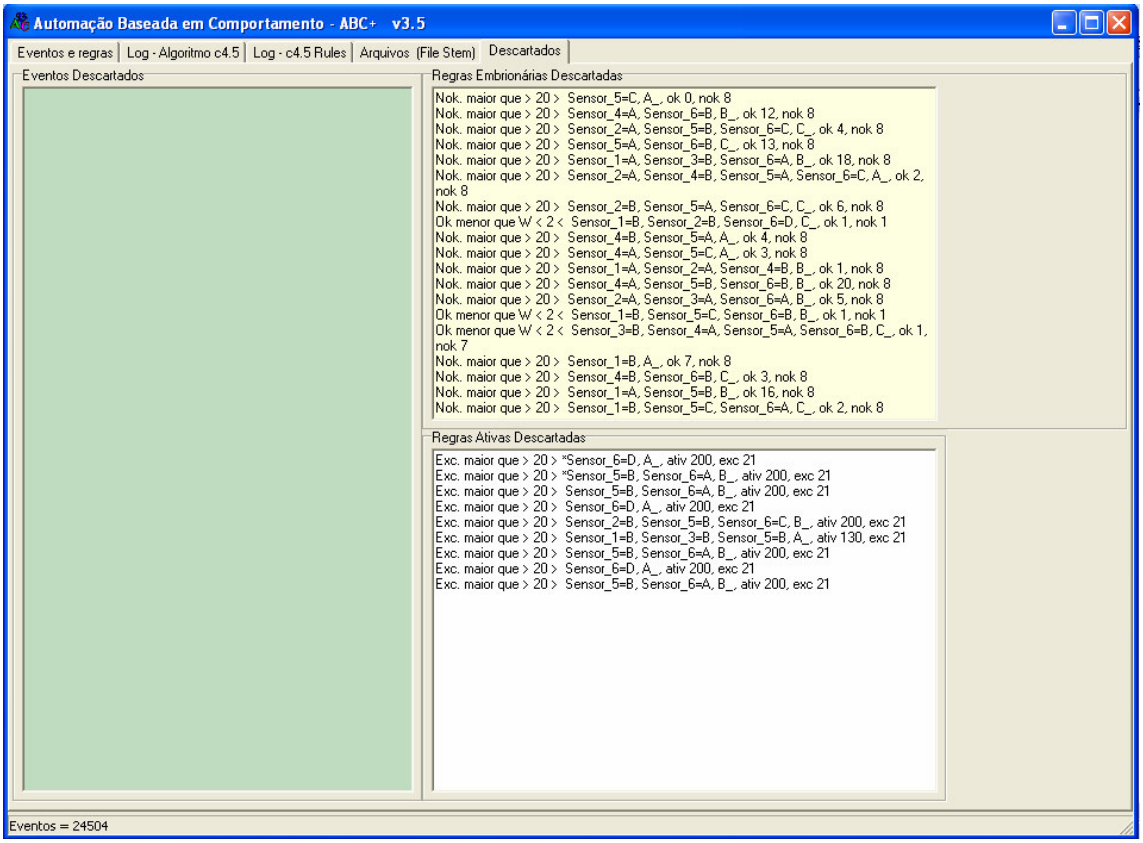


Figura 5.4 – Simulador para teste das variáveis do sistema ABC+ (Descartados).

A janela de observação não foi ativada nesta simulação e será avaliada em estudos e análises futuras. A avaliação de eventos descartados pela janela de observação deve ser feita com Banco de Dados representando a rotina real de uma residência. As simulações com o banco de eventos não determinísticos não reflete as condições em que a janela de observação deve atuar. Importante notar que, tendo o Banco de Dados com a rotina real, é possível avaliar

qual a média dos valores de tempo a serem utilizados na janela de observação, ou seja, é o banco que definirá os valores de tempo para a janela de observação.

5.2.1 Parâmetros para Avaliação

O sistema atua criando e eliminado regras, tanto Regras Embrionárias quanto Regras Ativas. Quando é feita a manipulação dos valores das variáveis as quantidades de regras varia, ou seja, são criadas e eliminadas mais ou menos regras de acordo com a configuração das variáveis. Com base nisto, definiu-se que os parâmetros utilizados para a avaliação, os quais foram desenvolvidos no simulador, são:

- EC = Número de Regras Embrionárias criadas;
- AC = Número de Regras Ativas criadas;
- EE = Número de Regras Embrionárias eliminadas;
- AE = Número de Regras Ativas eliminadas (rebaixadas).

Com estes parâmetros é possível fazer uma análise quantitativa dos resultados de acordo com as variações que forem feitas.

Outra informação que o simulador fornece são quais regras e em qual momento da simulação as mesmas foram criadas e eliminadas, permitindo uma análise qualitativa.

5.3 Agente para Simulação

Para as simulações a serem feitas com o sistema ABC+, existe a necessidade de criar um agente para simulação. Este agente deve gerar eventos como um habitante em uma residência, os quais fazem o sistema ABC+ atuar armazenando eventos, criando regras, acionando regras, eliminando regras e realizando outras ações possíveis. A seguir é feita uma descrição sobre agentes e como deve ser um modelo de usuário; estas descrições servem para orientar a construção do agente para simulação.

5.3.1 Agentes

A definição de agentes vem sendo amplamente discutida, inclusive em diversas áreas como Inteligência Artificial, Sistemas de Informação, Computação Gráfica, entre outras, entretanto não se encontra um consenso sobre a mesma.

Uma definição bastante abrangente é dada por Russell e Norvig (2004):

"Um agente é tudo o que pode ser considerado capaz de perceber seu ambiente por meio de sensores e de agir sobre este ambiente por intermédio de atuadores."

A figura 5.5 apresenta a representação de um agente.

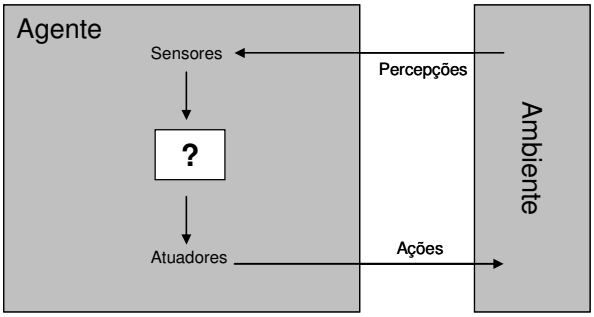


Figura 5.5 – Representação de um agente (RUSSELL; NORVIG, 2004).

Em sua definição, Wooldridge e Rao (1999) dizem que:

"Um agente é um sistema de computação que é capaz de agir independentemente (autônomo) de um usuário...".

Aprimora-se está definição destacando que o agente segue um propósito ou objetivo, fazendo com que suas ações modifiquem o ambiente conforme suas necessidades (CAVALHIERI, 2006).

O agente interage com outro agente ou com um humano por meio de uma linguagem de comunicação, exibe um comportamento através de objetivos, escolhe de maneira dinâmica que ações executar e em qual ordem (SANTOS, 2004).

5.3.2 Modelo de usuário

Um modelo de usuário conforme Santos (2004) é:

"...uma representação explícita das características, preferências e necessidades de um usuário".

Em outra definição segundo Kobsa (1995), o modelo de usuário é:

> *"...uma coleção de informações e suposições sobre usuários individuais ou sobre grupos de usuários, necessárias para que o sistema adapte diversos aspectos de suas funcionalidades e interface".*

Para modelar o usuário é imprescindível definir as informações necessárias para a elaboração do modelo. As tarefas realizadas pelo sistema, os objetivos do usuário em relação ao sistema e as características do sistema que serão afetadas pelo usuário devem ser consideradas na identificação dos propósitos da utilização do modelo.

5.3.3 Agentes Utilizados em Simulações

Vários trabalhos utilizam agentes para simular usuários ou ambientes. No trabalho de Bolzani (2004a) os eventos gerados pelos usuários estão previstos na forma de agentes que caminham por uma residência inteligente e interagem com ela. Além dos agentes usuários, existem agentes que simulam eventos de temperatura externa, umidade, meses do ano, dia/hora/minuto/segundo e eventos direto do integrador. Não existe uma definição clara de como os agentes foram criados, entretanto é informado que a variação dos agentes é feita através de controles deslizantes, ao gosto de quem está simulando o sistema. Para os eventos direto do integrador, o integrador gera os eventos a qualquer momento de maneira aleatória.

O trabalho de Domingues (2002) explica que uma das maneiras mais utilizadas para criação de dados para teste é a geração aleatória, na qual pontos do domínio dos dados são selecionados aleatoriamente. Não existe a garantia da melhor seleção, entretanto por sua facilidade de automatização e capacidade de gerar grandes conjuntos de dados de teste facilmente ela é defendida. A técnica também elimina a influência do testador na geração dos dados para teste.

Outro trabalho interessante que trabalha com a criação de agentes e ambientes usados em simulações é o trabalho de Moniz (1993). No trabalho a geração de eventos aleatórios é feita de acordo com um valor (gerado aleatoriamente) usado para selecionar uma ação de um banco de dados de ações.

Os trabalhos citados servirão de referência para a criação do agente de teste a ser utilizado neste trabalho.

5.3.4 Agente para simulação no sistema ABC+

O agente a ser utilizado nas simulações no sistema ABC+ possui o comportamento representado na figura 5.6.

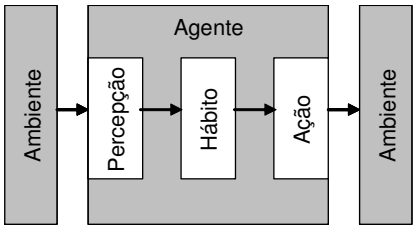


Figura 5.6 – Representação do agente no sistema ABC+.

Este agente para as simulações no sistema ABC+ deveria ter seus eventos definidos por um especialista, levando em consideração fatores tais como costumes, gostos, região e cultura, entre outros, do habitante que está simulando. Entretanto nas simulações a serem feitas serão abstraídos tais fatores, inclusive serão abstraídos os tipos de sensores e atuadores a serem utilizados. Isto se deve ao fato de que o que se deseja testar é o funcionamento do sistema ABC+, sendo irrelevante nesta etapa de simulação que o agente (habitante) e a residência tenham seus modelos exatamente iguais aos reais, o que seria extremamente difícil de se replicar.

O agente para simulação tem como objetivo unicamente gerar eventos. Os eventos gerados serão capazes de satisfazer as necessidades e restrições para os testes do funcionamento do sistema proposto. O agente gera eventos através da troca de estados dos sensores e atuadores de uma residência. Aos sensores são aplicadas restrições, as quais o agente deve respeitar. Devem ser seguidas também as preferências do habitante, as quais o agente irá representar através de regras entre os atuadores e sensores. As regras do agente são padrões repetitivos de comportamento. Ademais das restrições e preferências, o agente deve gerar eventos aleatórios para simular eventos esporádicos que normalmente acontecem em uma residência.

5.3.5 Banco de Dados para Simulação

Os eventos para simulação estão armazenados em um Banco de dados (BD) externo ao simulador. Estes eventos simulam a ação de um agente, cuja finalidade é simular um habitante real de uma casa e assim permitir estudar as variáveis do sistema. Este BD representa um mês de eventos.

O agente gerador de eventos, Banco de Dados, foi desenvolvido respeitando o descrito nos itens 5.3.3 e 5.3.4 deste trabalho. Serão consideradas as características, preferências e necessidades do ambiente e usuário (SANTOS, 2004). Ademais, será utilizada maneira de criação aleatória dos dados (DOMINGUES, 2002) (MONIZ, 1993).

Para o Banco de Dados foram considerados:

- Seis Sensores (de Sensor1 a Sensor6) sendo Sensor1 com estados A e B, Sensor2 com estados A e B, Sensor3 com estados A e B, Sensor4 com estados A e B, Sensor5 com estados A, B e C e Sensor6 com estados A, B, C e D.
- Um atuador (Atuador) com os estados A, B e C.

Os sensores são o ambiente ao qual o agente para simulação tem percepção, ou seja, ele observa o estado dos sensores. O atuador é o ambiente ao qual o agente para simulação realiza a ação, ou seja, ele atua definindo o estado do atuador; isto pode ser observado na figura 5.6.

Para criação do BD foram feitas combinações com os estados dos sensores e atuador. Com os sensores considerados (Sensor1 a Sensor6) e seus devidos estados é possível ter 192 combinações (2x2x2x2x3x4). Exemplos de combinações podem ser observados na tabela 5.1.

Tabela 5.1 – Exemplos de combinação no Banco de Dados.

Atuador	Sensor1	Sensor2	Sensor3	Sensor4	Sensor5	Sensor6
A	A	B	A	B	C	D
B	B	A	B	A	A	C
C	A	A	A	B	B	A

Posteriormente foram consideradas, como uma premissa, condições que não podem acontecer entre os sensores. Isto foi feito para diminuir a quantidade de combinações a serem trabalhadas no Banco de Dados. Um exemplo prático de combinações que não podem acontecer pode ser a existência de um sensor de temperatura e outro de umidade, neste caso pode-se considerar, somente como premissa, que nunca existirá a condição de temperatura alta e umidade alta ao mesmo tempo. Para o banco de dados as premissas são:

- (Sensor2=A E Sensor5=C) OU
- (Sensor3=B E Sensor6=D) OU
- (Sensor1=B E Sensor4=A).

Com estas condições, existem 105 combinações possíveis de acontecer e 87 que não podem acontecer.

Em seguida foram considerados 5 padrões do agente para simulação, padrões estes refletidos em regras de automação que devem estar contidas no BD:

- R1: SENSOR1=A E SENSOR2=B E SENSOR6=D ENTÃO ATUADOR=A;
- R2: SENSOR2=B E SENSOR3=A E SENSOR5=B ENTÃO ATUADOR=B;
- R3: SENSOR3=A E SENSOR4=B E SENSOR6=B ENTÃO ATUADOR=C;
- R4: SENSOR1=B E SENSOR3=B E SENSOR5=B ENTÃO ATUADOR=A;
- R5: SENSOR2=A E SENSOR3=B E SENSOR5=B E SENSOR6=A ENTÃO ATUADOR=B.

As regras são os hábitos do agente para simulação, os quais para cada posição dos sensores (percepção) ele indica uma posição (ação) de atuador (fig. 5.6).

A partir das regras foram encontradas 32 combinações de sensores que as geram. Das 105 combinações possíveis, retirando-se estas 32 combinações citadas anteriormente restam 73 combinações, para as quais não existem padrões de comportamento, portanto não existe estado de Atuador determinado para estas combinações. Para as cinco regras estipuladas, não há no BD nenhuma combinação que as contrarie.

Na realidade, em 5 das 32 combinações, o Atuador poderia ter assumido dois valores, isto acontece porque a combinação dos sensores satisfaz a mais do que uma única regra desejada (R1 a R5). A Tabela 5.2 apresenta uma destas combinações. Na primeira linha a regra um está sendo atendida e na segunda linha a regra dois está sendo atendida, para uma mesma combinação nos sensores. Neste caso definiu-se que a regra um é a que seria atendida no Banco de Dados. Isto implica que para a regra dois uma de suas combinações não irá aparecer e o algoritmo C4.5 terá de generalizar a regra sem esta combinação. Note que quando a regra dois for criada, toda vez que esta combinação aparecer no Banco de Dados, a regra será contrariada, pois o valor do Atuador será A e não B. Com isto deverão ser observados dois pontos: A regra dois, ou as outras que também estão na mesma condição, aparecerá tal como foi escrita ou estará mais especializada em função de uma de suas combinações não existir? As contradições que a regra dois irá sofrer durante as simulações farão com que a mesma desapareça ou permaneça no BDEmbrio ou BDAtivas?

Tabela 5.2 – Exemplo de Combinação que atende duas regras.

Atuador	Sensor1	Sensor2	Sensor3	Sensor4	Sensor5	Sensor6
A	A	B	A	A	B	D
B	A	B	A	A	B	D

Para cada uma das 32 combinações foi considerado que existe certa freqüência mensal de repetição. O uso de uma freqüência de repetição serve para representar quantas vezes no mês uma determinada combinação irá acontecer, isto se refletirá no Banco de Dados em várias posições de memória (linhas), contendo a mesma informação em todas elas.

As combinações têm freqüência mensal (repetições) de valores igual a 180 (para 6 combinações), 90 (para 6 combinações), 30 (para 8 combinações), 15 (para 6 combinações) e 3 (para 6 combinações), conforme a tabela 5.3. Estas repetições geram um total de 1.968 (180x6+90x6+30x8+15x6+3x6) eventos (posições de memória do BD).

As outras 73 combinações possíveis foram copiadas 15 vezes, gerando mais 1.095 (73x15) eventos. Para estes eventos o valor do estado do Atuador foi produzido aleatoriamente, com isto têm-se as 73 combinações sem valor de atuador padronizado. Estes eventos são eventos esporádicos (serão assim chamados) e não é desejável que eles produzam regras, por este motivo é que seus valores foram produzidos aleatoriamente. Ao final existem 3.063 eventos no BD.

Tabela 5.3 – Combinações das regras e suas freqüências.

Sensor1	Sensor2	Sensor3	Sensor4	Sensor5	Sensor6	Atuador1	Freqüência Mensal	Regra
A	B	A	A	A	D	A	180	1
A	B	A	A	B	D	A	180	
A	B	A	A	C	D	A	90	
A	B	A	B	A	D	A	90	
A	B	A	B	C	D	A	30	
B	A	B	B	B	A	A	30	4
B	A	B	B	B	B	A	30	
B	A	B	B	B	C	A	15	
B	B	B	B	B	A	A	15	
B	B	B	B	B	B	A	3	
B	B	B	B	B	C	A	3	
A	A	B	A	B	A	B	180	5
A	A	B	B	B	A	B	180	
A	B	A	A	B	A	B	90	2
A	B	A	A	B	B	B	90	
A	B	A	A	B	C	B	30	
A	B	A	B	B	A	B	30	
A	B	A	B	B	C	B	30	
A	B	A	B	B	D	B	15	
B	B	A	B	B	A	B	15	
B	B	A	B	B	C	B	3	
B	B	A	B	B	D	B	3	
A	A	A	B	A	B	C	180	3
A	A	A	B	B	B	C	180	
A	B	A	B	A	B	C	90	
A	B	A	B	B	B	C	90	
A	B	A	B	C	B	C	30	
B	A	A	B	A	B	C	30	
B	A	A	B	B	B	C	15	
B	B	A	B	A	B	C	15	
B	B	A	B	B	B	C	3	
B	B	A	B	C	B	C	3	

Tabela 5.4 – Exemplo de embaralhamento (eventos desordenados).

Sensor1	Sensor2	Sensor3	Sensor4	Sensor5	Sensor6	Atuador1	Aleatório
A	B	A	A	A	D	A	76
A	B	A	A	B	D	A	27
A	B	A	A	C	D	A	31
A	B	A	B	A	D	A	23
A	B	A	B	C	D	A	7
B	A	B	B	B	A	A	90
B	A	B	B	B	B	A	65
B	A	B	B	B	C	A	8
B	B	B	B	B	A	A	20
B	B	B	B	B	B	A	46
B	B	B	B	B	C	A	47
A	A	B	A	B	A	B	100
A	A	B	B	B	A	B	90
A	B	A	A	B	A	B	82
A	B	A	A	B	B	B	92
A	B	A	A	B	C	B	43
A	B	A	B	B	A	B	81

Por último foram gerados números aleatórios entre 1 e 100 para cada um dos 3.063 eventos, como no exemplo da tabela 5.4. Os eventos foram ordenados pelos seus números aleatórios e assim ficaram embaralhados (tabela 5.5). O objetivo do embaralhamento é não ter um BD viciado, ou seja, ter os eventos distribuídos de maneira não determinística.

Tabela 5.5 – Exemplo de embaralhamento (eventos ordenados).

Sensor1	Sensor2	Sensor3	Sensor4	Sensor5	Sensor6	Atuador1	Aleatório
A	B	A	B	C	D	A	7
B	A	B	B	B	C	A	8
B	B	B	B	B	A	A	20
A	B	A	B	A	D	A	23
A	B	A	A	B	D	A	27
A	B	A	A	C	D	A	31
A	B	A	A	B	C	B	43
B	B	B	B	B	B	A	46
B	B	B	B	B	C	A	47
B	A	B	B	B	B	A	65
A	B	A	A	A	D	A	76
A	B	A	B	B	A	B	81
A	B	A	A	B	A	B	82
B	A	B	B	B	A	A	90
A	A	B	B	B	A	B	90
A	B	A	A	B	B	B	92
A	A	B	A	B	A	B	100

Obtém-se um Banco de Dados com 3.063 eventos embaralhados, simulando a ação de um agente, a qual é não determinística. Existem 192 combinações de sensores, das quais 87 estão eliminadas, 32 combinações estão vinculadas a 5 padrões e 73 combinações de sensores não geram padrões específicos.

Foram gerados outros dois bancos de dados com os mesmos eventos, porém embaralhados diferentemente. Isto permite que as simulações sejam feitas com três bancos de dados, sendo considerado como resultado sempre a média das três simulações.

5.4 Resultados Obtidos

Foram feitas variações dos valores de cada variável em análise na realização das simulações. Em cada variação foram feitas três simulações, uma para cada BD de eventos, e foram obtidos os dados dos parâmetros. Em algumas simulações também foram colhidos dados mais detalhados, para se observar quais regras e quando as mesmas foram produzidas. Foram realizadas análises quantitativas, baseadas nos valores dos parâmetros encontrados e análise qualitativa, para a qual foram utilizados os traçados das simulações.

5.4.1 Análises Quantitativas

A partir dos dados dos parâmetros obtidos nas simulações foram criados os gráficos das figuras 5.7 a 5.13. Cada gráfico representa uma variável do sistema em análise. O eixo Y dos gráficos sempre representa a quantidade de regras e o eixo X de cada gráfico a variável que está sendo estudada.

Foram feitas várias simulações com variações significativas dos valores das variáveis; muitas destas simulações apresentaram resultados que não permitem chegar a qualquer conclusão, entretanto quando foi encontrada a gama certa de variação, os resultados obtidos foram bastante satisfatórios e contribuíram muito para as conclusões.

A primeira observação importante é que os eventos esporádicos, que estão inseridos no BD, acabaram gerando e contrariando novas regras, como resultado o número de regras excluídas, tanto no BDEmbrio (EE) quanto no BDAtivas (AE), foi aumentado. Um aspecto importante é que os eventos esporádicos são benéficos ao sistema, isto porque tornam o trabalho do C4.5 mais consistente para o sistema. Na inexistência dos eventos esporádicos o C4.5 cria regras muito genéricas, como será possível verificar no item 5.4.2.

Como segunda observação, não foram geradas somente as 5 regras desejadas, mas sim muitas outras. As regras geradas, sem contar as regras indesejáveis provenientes dos eventos esporádicos, foram sempre regras relacionadas com as 5 regras desejadas, entretanto não são sempre exatamente as desejadas, mas sim regras especializadas ou generalizadas demais em relação as originais. Com o surgimento de regras especializadas ou generalizadas, os eventos

do BD subseqüentes a tais regras acabaram gerando várias exclusões das mesmas, fato que também contribuiu muito para o aumento de regras excluídas (EE e AE).

5.4.2 Avaliação da Variável NuEv

O gráfico da figura 5.7 mostra o comportamento do sistema quando é feita a variação de NuEv, quantidade de eventos inseridos no C4.5.

As simulações foram feitas variando o valor de NuEv desde 20 até 500, mantendo os seguintes valores das outras variáveis:

- OK_Emb = 20;
- NOK_Emb = 7;
- EXC_At = 20;
- ATIV_At = 2;
- OK_Emb_exc = 2.

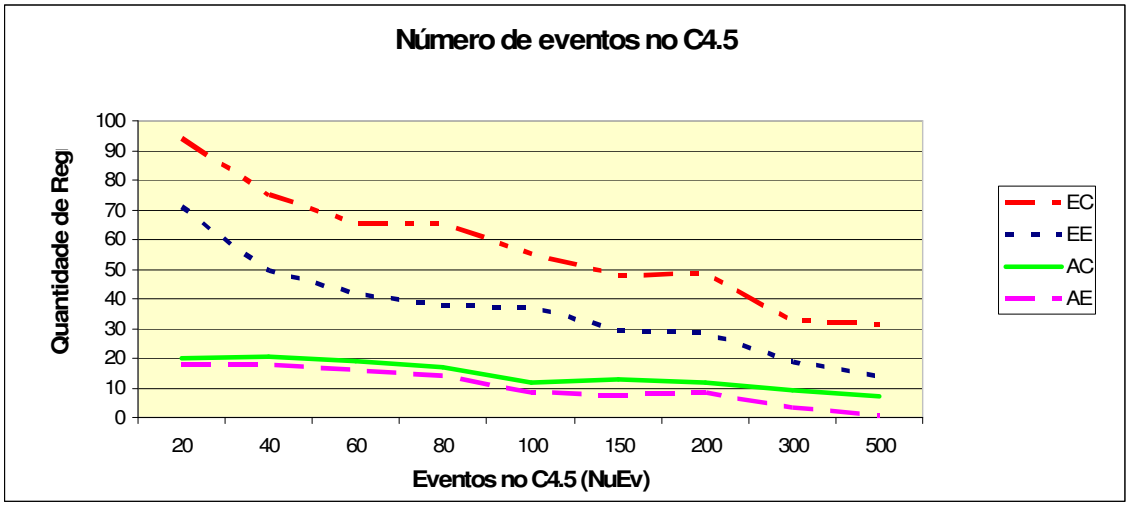


Figura 5.7 – Gráfico da variação de NuEv.

Observa-se pelo gráfico que todos os parâmetros de quantidades de regras decrescem com o aumento do número de eventos. Este resultado era o esperado para esta variável, isto acontece devido ao fato de que quanto menos eventos se coloca no C4.5, mais regras que não deveriam ser criadas (generalizadas, especializadas e esporádicas ou indesejáveis) surgem. Quando regras são criadas erroneamente, os eventos subseqüentes às mesmas no BD acabam excluindo-as; com isto novos eventos vão para o BDEventos e acionam novamente o C4.5 para criar novas regras, repetindo-se novamente o ciclo até o fim do eventos. Quando se aumenta o número de eventos no C4.5, as regras geradas tendem a ser mais parecidas com as

regras desejadas, ocorrendo menos exclusões e menos criações de novas regras. De fato, algumas das 5 regras definidas começam a aparecer com NuEv em 20. Todas as 5 regras definidas aparecem, simultaneamente, com o valor de 300 para NuEv, sendo portanto este o melhor valor da variável para a configuração das simulações.

O gráfico da figura 5.8 foi feito com outros três Bancos de Dados onde os eventos esporádicos foram suprimidos, ou seja, foram colocados apenas eventos que geram as 5 regras desejadas. O que se observou foi a severa diminuição em todos os parâmetros de análise, isto em função de não existir eventos que ficassem contrariando as regras que foram geradas. Um fato observado foi que as regras ficaram extremamente generalizadas, sendo que em nenhum momento, com nenhum valor de NuEv, foram encontradas exatamente as regras desejadas. Isto demonstra que o C4.5 necessita de exemplos variados para gerar regras o mais específico possível.

Num caso real a presença de eventos esporádicos, os quais não representam a rotina mais específica do habitante, será de grande valor para que as regras corretas surjam no sistema. Apesar das regras geradas com este novo Banco de Dados serem genéricas, o sistema cumpriu o seu objetivo, visto que regras foram criadas e passaram a atuar acionando o atuador, o problema é que os eventos ficaram confinados em uma gama pequena de combinações, perante todas as possíveis, neste caso se começassem a surgir outras combinações de eventos as regras criadas poderiam passar a ser contrariadas. Enfim, é importante que se tenha uma boa variedade de exemplos para que o algoritmo C4.5 possa criar regras especificas, condizentes com os atributos (sensores) que são manipulados na rotina do habitante da casa.

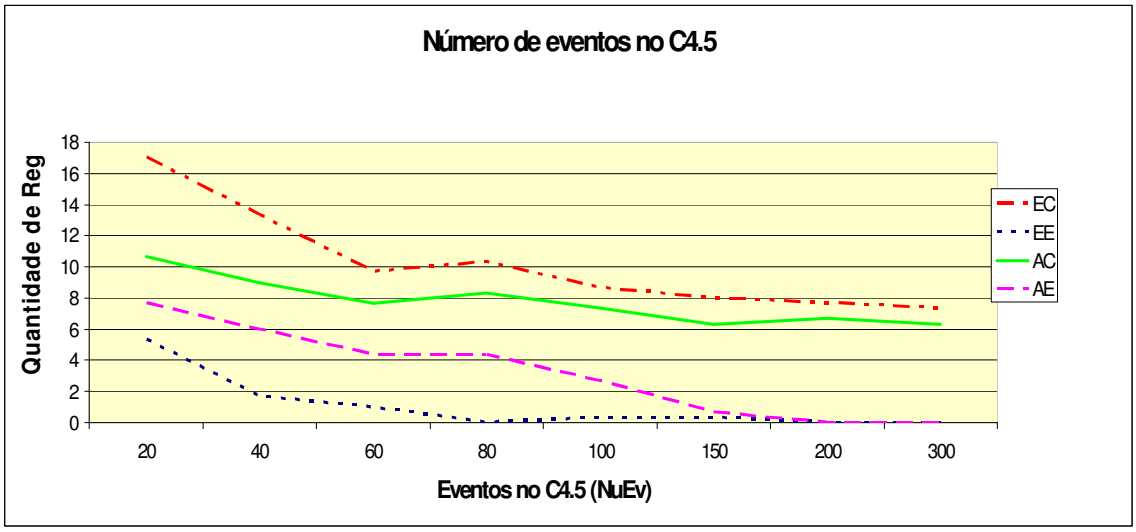


Figura 5.8 – Gráfico da variação de NuEv, com BD somente com regras desejadas.

5.4.3 Avaliação da Variável OK_Emb

A figura 5.9 apresenta o gráfico da variação de OK_Emb, valor máximo da variável OK. Esta variável define quantas vezes a Regra Embrionária necessita ser confirmada antes de a mesma ser promovida à Regra Ativa.

As simulações foram feitas variando o valor de OK_Emb desde 5 até 30, mantendo os seguintes valores das outras variáveis:

- NuEv = 200;
- NOK_Emb = 7;
- EXC_At = 20;
- ATIV_At = 2;
- OK_Emb_exc = 2.

Observa-se que quanto maior for o valor desta variável menos Regras Embrionárias são promovidas e conseqüentemente menos Regras Ativas serão criadas, isto ocorre porque a Regra Embrionária tem de receber uma pontuação maior para ser promovida, o que torna mais difícil sua promoção. Como as Regras Embrionárias não são promovidas, elas acabam sendo eliminadas, daí o aumento desta eliminação no gráfico. Estas eliminações acabam diminuindo o número de regras indesejáveis no sistema.

O comportamento desta variável é o que se esperava, pois o objetivo da variável é facilitar ou dificultar a promoção da Regra Embrionária a Regra Ativa.

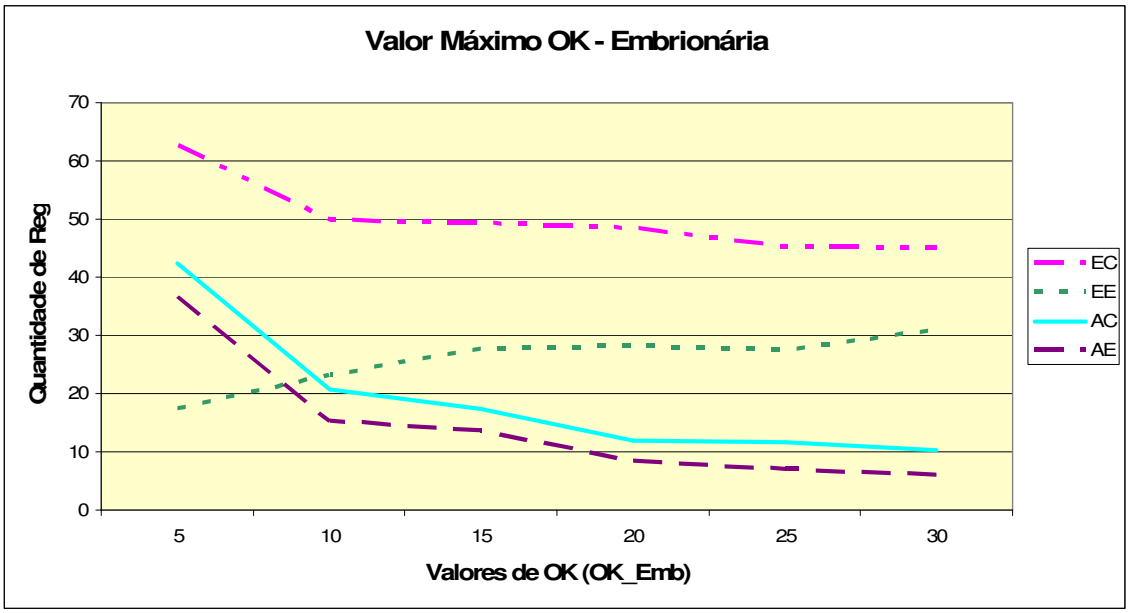


Figura 5.9 – Gráfico da variação de OK_Emb.

O cruzamento da linha EE com as linhas AC e AE no gráfico representa algo importante, é a partir deste ponto (OK_Emb maior ou igual a 10) que existem mais Regras Embrionárias excluídas do que Regras Ativas criadas, ou seja, as Regras Embrionárias indesejadas são excluídas antes de virarem Regras Ativas, o que é desejável. Com OK_Emb igual a 20 já é possível verificar uma melhor estabilização nos parâmetros, sendo portanto este um bom valor encontrado para a variável.

5.4.4 Avaliação da Variável NOK_Emb

A variação de NOK_Emb, valor máximo da variável NOK, o qual define quantas vezes a Regra Embrionária pode ser contrariada antes de a mesma ser eliminada, é apresentada no gráfico da figura 5.10.

As simulações foram feitas variando o valor de NOK_Emb desde 5 até 30, mantendo os seguintes valores das outras variáveis:

- NuEv = 200;
- OK_Emb = 20;
- EXC_At = 20;
- ATIV_At = 2;
- OK_Emb_exc = 2.

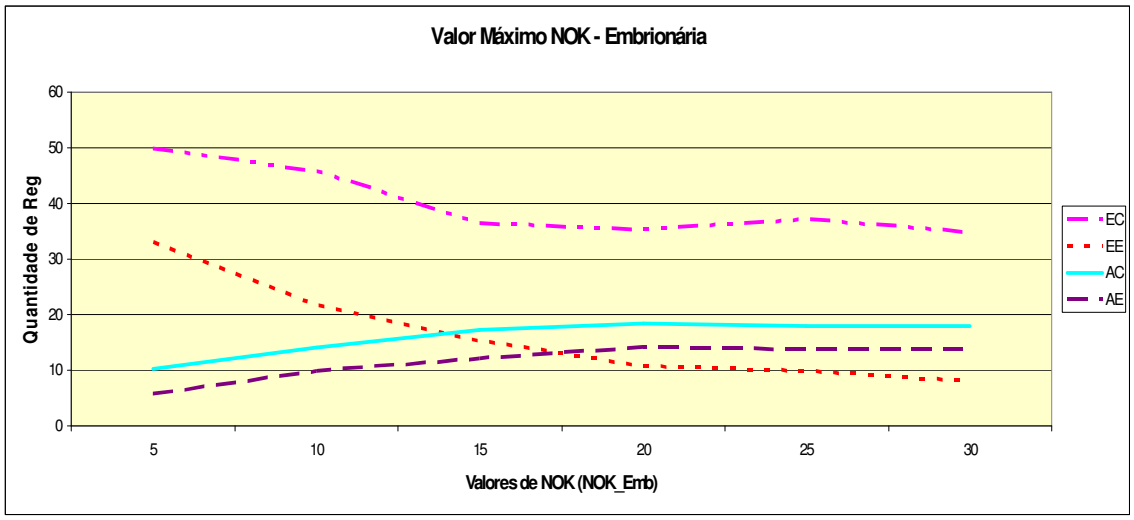


Figura 5.10 – Gráfico da variação de NOK_Emb.

Pelo gráfico observa-se que quanto maior o valor de NOK_Emb menos Regras Embrionárias são eliminadas e conseqüentemente mais Regras Ativas são criadas, ou seja, é dificultada a eliminação da Regra Embrionária e facilitada a promoção da mesma para Regra Ativa. Nas simulações feitas, a partir do valor de NOK_Emb igual a 20, se observa uma estabilidade nos dados, ou seja, já não existe a facilitação citada, pois não basta dificultar a eliminação da Regra Embrionária para a mesma ser promovida a Ativa, torna-se necessário também que está regra ganhe pontos positivos para poder acender a Regra Ativa (variável OK_Emb). Como o objetivo de NOK_Emb é facilitar ou dificultar a eliminação das Regras Embrionárias, a variável teve o comportamento esperado.

O cruzamento da linha EE com as linhas AC e AE no gráfico representa novamente algo importante, é a partir deste ponto (NOK_Emb maior do que 10) que existem menos Regras Embrionárias excluídas do que Regras Ativas criadas, ou seja, as Regras Ativas surgem antes das Regras Embrionárias terem sido excluídas, o que é indesejável. Com OK_Emb menor do que 10 existem menos Regras Ativas criadas do que Regras Embrionárias excluídas, sendo portanto este o valor a ser utilizado nesta variável.

5.4.5 Avaliação da Variável EXC_At

No gráfico da figura 5.11 ocorre a variação no valor de EXC_At, valor máximo da variável EXC, o qual define quantas vezes a Regra Ativa pode ser contrariada antes da mesma ser rebaixada a Regra Embrionária.

As simulações foram feitas variando o valor de EXC_At desde 5 até 40, mantendo os seguintes valores das outras variáveis:

- NuEv = 200;
- OK_Emb = 20;
- NOK_Emb = 7;
- ATIV_At = 2;
- OK_Emb_exc = 2.

Observa-se que todos os parâmetros de quantidades de regras decrescem com o aumento do valor de EXC e a partir do valor da variável igual a 25 decrescem mais suavemente. A diminuição do número de Regras Embrionárias ocorre devido a que quanto maior o valor de EXC é mais difícil a Regra Ativa ser excluída e, por conseqüência, se não é excluída, não é novamente criada como Regra Embrionária. O fato de regras não serem excluídas também faz com que os eventos subseqüentes do Banco de Dados sejam utilizados

para pontuar as regras e não são armazenados no BDEventos para gerar novas regras, em outras palavras, um determinado evento irá gerar mais um ponto na campo EXC de uma Regra Ativa, caso esta regra já tivesse sido eliminada o evento em questão seria armazenado no BDEventos e acabaria gerando nova Regra Embrionária. Um ponto importante é que regras desejadas não são eliminadas, mesmo com valores maiores de EXC_At; isto pode ser observado a partir do valor de EXC_At igual a 20, onde a quantidade de Regras Ativas se mantém praticamente constante.

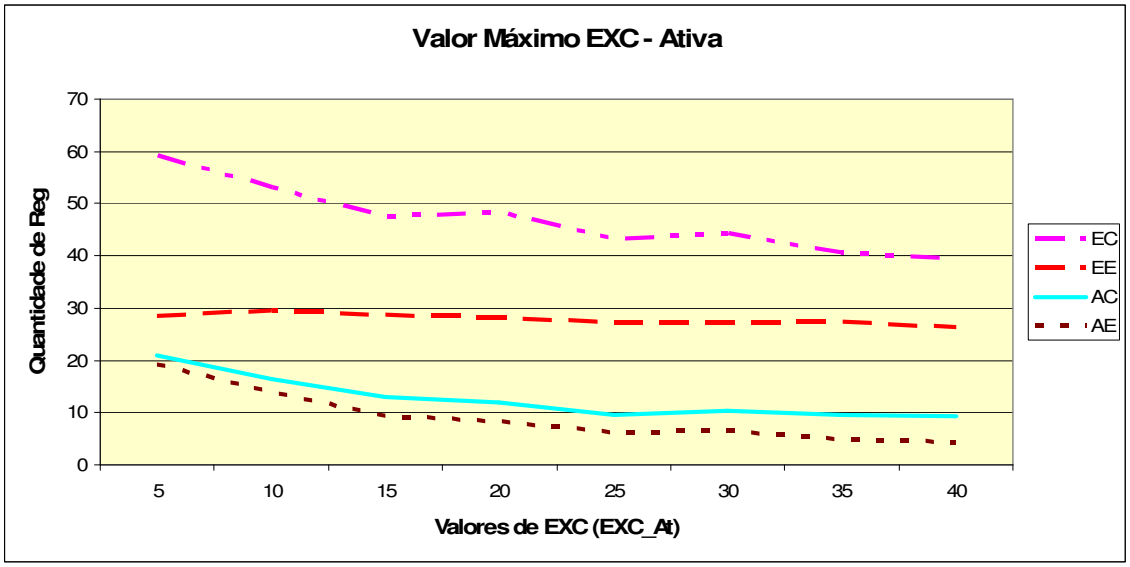


Figura 5.11 – Gráfico da variação de EXC_At.

5.4.6 Avaliação da Variável ATIV_At

No gráfico da figura 5.12 ocorre a variação no valor de ATIV_At, valor mínimo da variável ATIV, cujo valor determina o mínimo de vezes que uma Regra Ativa tem de ser acionada entre cada atuação do C4.5 para não ser rebaixada à Regra Embrionária por desuso.

As simulações foram feitas variando o valor de ATIV_At desde 1 até 300, mantendo os seguintes valores das outras variáveis:

- NuEv = 200;
- OK_Emb = 5;
- NOK_Emb = 50;
- EXC_At = 500;
- OK_Emb_exc = 1.

Pelo gráfico observa-se que quanto maior o valor de ATIV_At mais Regras Ativas são eliminadas, ou seja, é dificultada a permanência da Regra Ativa. Observou-se também que com o aumento de ATIV_At surge um número maior de Regras Embrionárias e Regras Ativas criadas; isto acontece porque quando as Regras Ativas são rebaixadas aumenta o número de Regras Embrionárias criadas, por outro lado algumas destas regras são novamente promovidas a Regras Ativas e assim aumentando o número de Regras Ativas criadas.

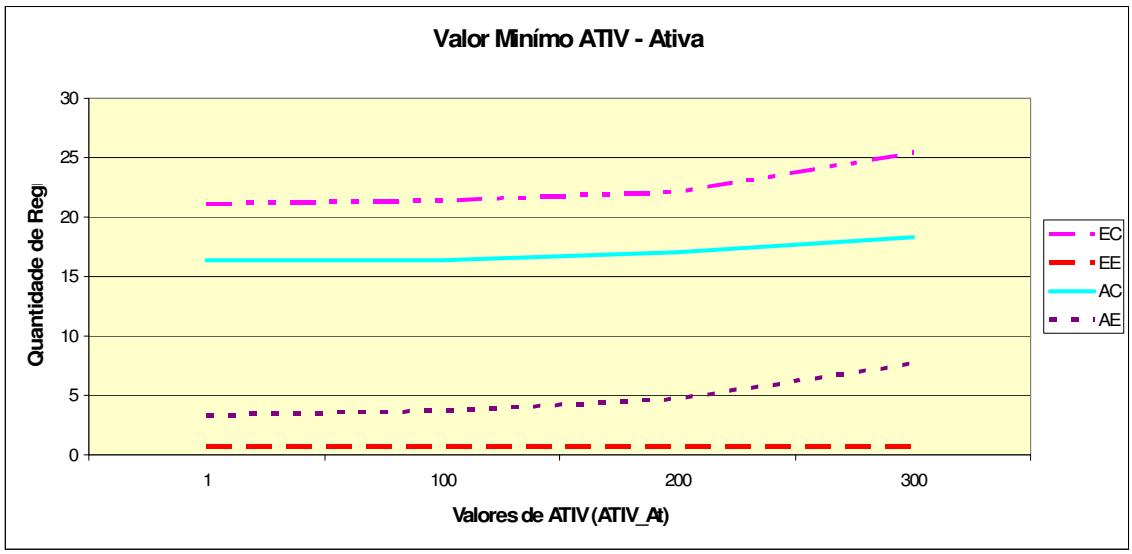


Figura 5.12 – Gráfico da variação de ATIV_At.

Os valores de ATIV são altos, isto acontece devido à maneira que o Banco de Dados de testes foi criado, onde um evento deste Banco de Dados não tem relação com o evento posterior. Neste caso um novo evento atua como se fossem até seis mudanças de sensores na residência, o que causa a repetição por seis no valor de ATIV quando existir atuação neste campo, ou seja, quando uma Regra Ativa existir e receber um ponto positivo, na verdade recebe até seis pontos. A variável ATIV_At é realmente uma variável de controle de desuso das Regras Ativas, sendo utilizada na prática somente para garantir que a Regra Ativa que caiu em desuso seja um dia rebaixada a Regra Embrionária. A variação da variável não apresentou grande variação no sistema.

5.4.7 Avaliação da Variável OK_Emb_exc

No gráfico da figura 5.13 ocorre a variação no valor de OK_Emb_exc, valor mínimo da variável OK, cujo valor determina o mínimo de vezes que uma Regra Embrionária tem de ser acionada entre cada atuação do C4.5 para não ser eliminada por desuso.

As simulações foram feitas variando o valor de OK_Emb_exc desde 1 até 17, mantendo os seguintes valores das outras variáveis:

- NuEv = 100;
- OK_Emb = 20;
- NOK_Emb = 20;
- EXC_At = 20;
- ATIV_At = 2.

Pelo gráfico observa-se que quanto maior o valor de OK_Emb_exc mais Regras Embrionárias são eliminadas, ou seja, é dificultada a permanência da Regra Embrionária. Observou-se também que com o aumento de OK_Emb_exc surge um número maior de Regras Embrionárias criadas; isto acontece porque quando as Regras Embrionárias são eliminadas, os eventos subseqüentes às mesmas no BD acabam sendo armazenados no BDEventos, acionando novamente o C4.5 e criando novas Regras Embrionárias, repetindo-se novamente este ciclo até o fim do eventos.

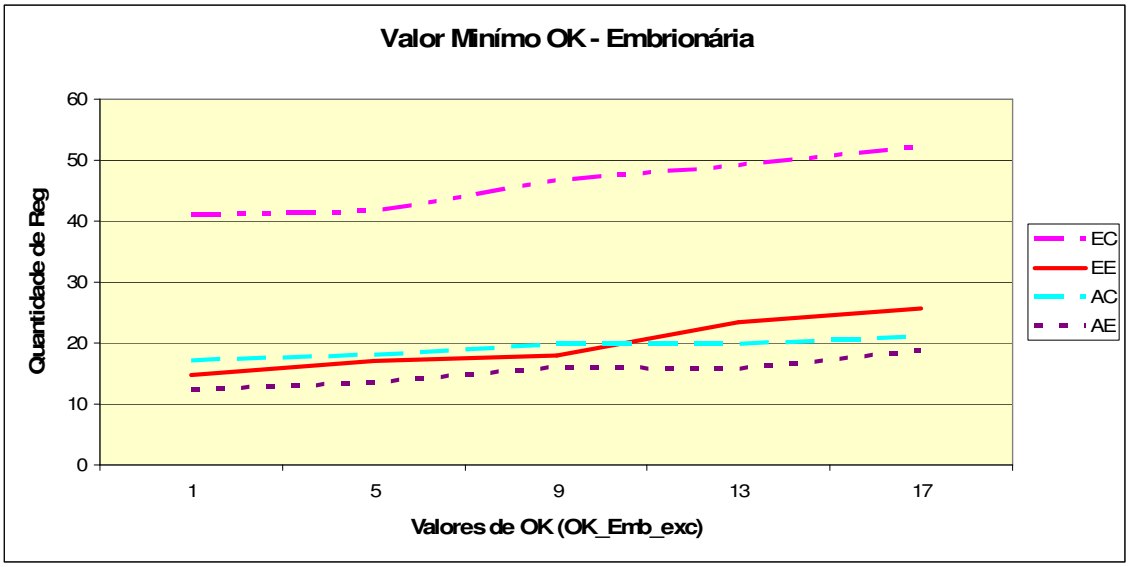


Figura 5.13 – Gráfico da variação de OK_Emb_exc.

O cruzamento da linha EE com a linha AC no gráfico representa que tanto as Regras Embrionárias excluídas quanto as Regras Ativas criadas crescem com o crescimento no número de Regras Embrionárias criadas, entretanto em dado momento mais Regras Embrionárias são excluídas do que Regras Ativas são criadas, isto demonstra que para a configuração utilizada é mais fácil excluir uma Regra Embrionária do que promovê-la a Regra Ativa, sendo algo desejável no sistema. A variável OK_Emb_exc é realmente uma variável de controle de desuso das Regras Embrionárias, sendo utilizada na prática somente para garantir

que a Regra Embrionária que caiu em desuso seja um dia excluída do sistema. A variação da variável não apresentou grande variação no sistema

As variáveis observadas na análise quantitativa apresentaram o comportamento esperado, conforme descrito no capitulo 4, não houveram resultados que contrariassem o que foi proposto, nem que demonstrassem que o sistema apresenta falhas no seu funcionamento. Vale ressaltar que a análise está baseada na configuração desenvolvida (seis sensores, um atuador e o Banco de Dados de teste), sendo importante em trabalhos futuros avaliar novas configurações.

Como última consideração, a não limitação do número máximo de regras do BDAtivas não influenciou no processamento ou na capacidade de memória utilizados pelos simuladores. As quantidades de Regras Ativas existentes simultaneamente nos testes, sem limitação, não ultrapassou em nenhum momento o número de doze regras.

As tabelas com todos os valores das variáveis e resultados das simulações, os quais permitiram a confecção dos gráficos desta seção, são apresentadas no Apêndice 2.

5.4.8 Análise Qualitativa

A análise qualitativa está baseada na avaliação dos traçados (relatório detalhado da simulação). Com o traçado é possível observar cada evento do Banco de Dados sendo inserido no simulador e a respectiva ação do simulador. Sendo assim, pode-se verificar quando cada Regra Embrionária ou Regra Ativa é criada e quando é eliminada (ou rebaixada).

A tabela 5.6 apresenta um exemplo de parte de um traçado. O traçado deste exemplo foi retirado de uma simulação onde NuEv é igual a 20, ou seja, são necessários 20 eventos para acionar o algoritmo C4.5. Nele é possível observar na primeira coluna os eventos em seqüência que foram sendo gerados; sendo que a informação de cada linha (posição de memória) contém os dados dos sensores de 1 a 6 separados por virgula e mais o dado do atuador, o qual possui o símbolo *underline* "_" no final.

Na segunda coluna estão as Regras Embrionárias; no exemplo, após 20 eventos são criadas 6 Regras Embrionárias, para as quais os campos OK e NOK iniciam-se com valor zero. A cada novo evento é feita ou a pontuação de OK e NOK para as regras que estiverem vinculadas com o evento, ou a inserção do evento na coluna EVENTOS, no caso do evento não possuir nenhuma regra vinculada e ele. Quando uma Regra Embrionária se transforma em Regra Ativa, a mesma é colocada na terceira coluna. A regra aparece com os valores de ATIV e EXC inicialmente iguais a zero. No exemplo, após as Regras Embrionárias apresentadas,

foram colocadas duas linhas com informação "....." para expressar que vários eventos foram realizados até o surgimento de uma Regra Ativa.

Tabela 5.6 – Exemplo de Traçado.

EVENTOS	REGRAS EMBRIONARIAS	REGRAS ATIVAS
A, B, A, A, B, D, A_		
A, B, A, A, B, D, A_		
A, B, A, A, C, D, A_		
A, B, A, B, A, D, A_		
A, B, A, B, A, D, A_		
B, B, B, B, B, A, A_		
A, A, B, A, B, A, B_		
A, A, B, B, B, A, B_		
A, B, A, A, B, A, B_		
A, B, A, A, B, B, B_		
A, B, A, B, B, D, B_		
A, A, A, B, A, B, C_		
A, A, A, B, A, B, C_		
A, A, A, B, A, B, C_		
A, A, A, B, B, B, C_		
A, A, A, B, B, B, C_		
A, A, B, A, A, A, C_		
B, B, B, B, A, C, A_		
B, B, A, B, C, A, C_		
A, A, B, A, A, B, A_		
	Sensor_2=A, Sensor_3=A, C_, ok 0, nok 0	
	Sensor_6=D, A_, ok 0, nok 0	
	Sensor_5=B, B_, ok 0, nok 0	
	Sensor_5=B, B_, ok 0, nok 0	
	Sensor_6=D, A_, ok 0, nok 0	
	Sensor_2=A, Sensor_3=A, C_, ok 0, nok 0	
.....		
.....		
		Sensor_6=D, A_, ativ 0, exc 0

Foram colhidos traçados de todas as simulações feitas para a análise da variável NuEV (item 5.4.2), a menos do valor 500. Portanto foram obtidos traçados com valores de NuEv iguais a 20, 40, 60, 80, 100, 150, 200 e 300 para cada um dos três bancos de dados de teste.

A partir dos traçados foi feita a análise das 5 regras desejadas, as quais estão presentes no Banco de Dados de teste. As 5 regras e suas respectivas quantidades de repetições no Banco de Dados de teste são:

- R1: SENSOR1=A E SENSOR2=B E SENSOR6=D ENTÃO ATUADOR=A; 570 repetições.
- R2: SENSOR2=B E SENSOR3=A E SENSOR5=B ENTÃO ATUADOR=B; 306 repetições.
- R3: SENSOR3=A E SENSOR4=B E SENSOR6=B ENTÃO ATUADOR=C; 636 repetições.
- R4: SENSOR1=B E SENSOR3=B E SENSOR5=B ENTÃO ATUADOR=A;
- 96 repetições.
- R5: SENSOR2=A E SENSOR3=B E SENSOR5=B E SENSOR6=A ENTÃO ATUADOR=B; 180 repetições.

Foi avaliado se as regras realmente foram criadas e em que momento da simulação as mesmas são criadas, tanto como Regras Embrionárias quanto como Regras Ativas. O momento em que as regras surge está expresso em porcentagem. O Banco de Dados possui 3.063 eventos, que representam 100% do Banco de Dados. Uma porcentagem de 10% do Banco de Dados significa algo em torno de 306 eventos inseridos no simulador.

Pelo gráfico da figura 5.14 é possível observar que com NuEv igual a 20, surge nas simulações somente uma das regras desejadas, a regra 3. A regra 3 é a que mais repetições possui no Banco de Dados (636). A regra surge como Regra Embrionária quase no final da simulação (80%) e logo é promovida a Regra Ativa. Nos traçados de NuEv igual a 20, a maioria das regras criadas possui apenas um ou dois atributos e são regras generalizadas em relação as 5 regras desejadas, existem ainda regras oriundas dos eventos esporádicos. Importante ressaltar que a regra 3 é uma regra com três atributos. Estas generalizações ocorrem porque o algoritmo C4.5 tem poucos exemplos para criar as regras, o que consequentemente faz com que as mesmas sejam regras mais simplificadas (generalizadas).

Com o valor de NuEv igual a 40, surge novamente a regra 3, como Regra Embrionária e logo em seguida como Regra Ativa, entretanto ela aparece logo no início da simulação. Outra regra que surge no final da simulação é a regra 4, porém somente como Regra Embrionária. Para os traçados de NuEv igual a 40 valem os mesmos comentários de NuEv igual a 20, ou seja, as regras criadas possuem apenas um ou dois atributos e são regras generalizadas em relação as 5 regras desejadas, existem ainda regras oriundas dos eventos esporádicos

Para o valor de NuEv igual a 60 ocorre o mesmo que nos traçados anteriores, entretanto a regra criada é a regra 2.

As regras começam a surgir mais rapidamente a partir de NuEv igual a 80, onde surgem duas regras desejadas, as regras 2 e 3. Ambas regras surgem logo no início das simulações e em seguida a sua criação como Regras Embrionárias passam a Regras Ativas. Nestes traçados já é possível observar mais regras com dois e três atributos.

Os traçados de NuEv igual a 100 demonstram a existência de três regras desejadas (regra 2, 3 e 4). A maioria das regras possuem três atributos, ainda não é possível observar regras com quatro atributos, que é o caso da regra 5.

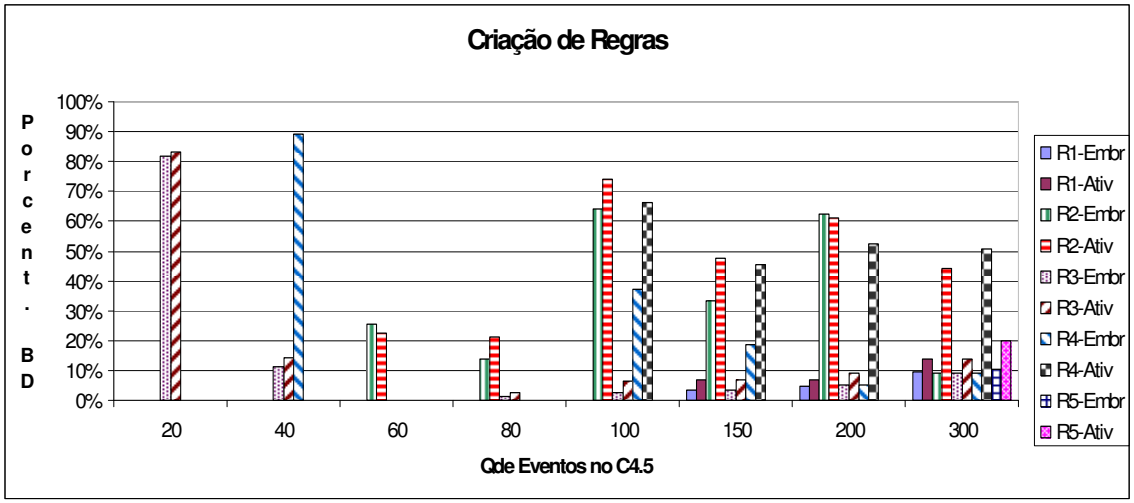


Figura 5.14 – Gráfico de Criação de Regras em função da quantidade de eventos no C4.5.

Com os valores de NuEv iguais a 150 e 200, surge a regra 1, além das outras três regras que já haviam surgido anteriormente. As Regras Embrionárias surgem logo no início das simulações e é possível observar regras com quatro atributos, porém a regra 5, que é a única desejada com quatro atributos, ainda não foi criada.

Efetivamente com 300 eventos sendo inseridos no algoritmo C4.5 (NuEv igual a 300), todas as 5 regras estão presentes. As regras surgem com cerca de 10% do Banco de Dados inserido no simulador, ou seja, como 10% são 306 eventos e o número de eventos no algoritmo C4.5 é 300, portanto logo na primeira execução do C4.5 todas as regras desejadas estão presentes, sendo promovidas a Regras Ativas na seqüência das simulações.

Pela análise quantitativa, onde podem ser observados os melhores valores de cada uma das variáveis nos itens 5.4.2 até 5.4.7, e pela análise qualitativa realizada, ambas baseadas na configuração desenvolvida (seis sensores, um atuador e o Banco de Dados de teste), os valores das variáveis que apresentaram o melhor resultado foram os seguintes:

- NuEv = 300;
- OK_Emb = 20;
- NOK_Emb = 7;
- EXC_At = 20;
- ATIV_At = 2;
- OK_Emb_exc = 2.

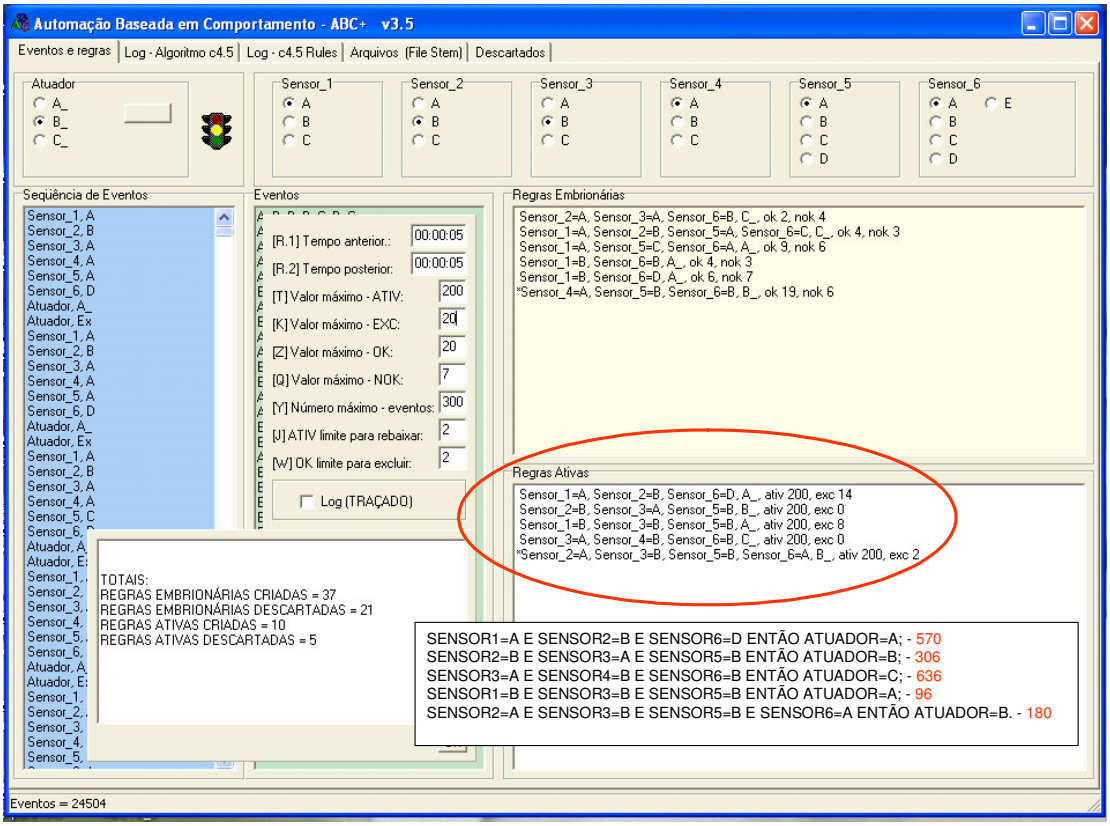


Figura 5.15 – Tela do simulador com resultados satisfatórios.

A figura 5.15 apresenta a tela do simulador, com os valores das variáveis descritos anteriormente, onde se pode observar no campo de Regras Ativas (elipse em vermelho) a presenças das 5 regras desejadas.

Com as análises realizadas é possível responder a duas questões feitas no decorrer do item 5.3.5 deste trabalho: A regra dois, ou as outras que também estão na mesma condição, aparecerá tal como foi escrita ou estará mais especializada em função de uma de suas combinações não existir? As contradições que a regra dois irá sofrer durante as simulações farão com que a mesma desapareça ou permaneça no BDEmbrio ou BDAtivas?

As respostas para estas questões são: A regra dois, e outras na mesma condição dela, por vezes apareceu mais especializada e as contradições fizeram com que a mesma desaparecesse do BDEmbrio ou BDAtivas, entretanto quando as variáveis do sistemas foram configuradas de certa maneira, a regra foi criada corretamente e mesmo sendo contrariada não foi eliminada dos bancos de regras.

A análise qualitativa permitiu observar que o sistema proposto pode ter suas variáveis configuradas de tal maneira a produzir regras que refletem a rotina desenvolvida pelo agente (habitante da casa).

Foi observado também através dos traçados que são criadas regras sobre especializadas, fato conhecido como *overfitting.* Um exemplo disto é a criação de regras com quatro atributos, quando a regra desejada deveria ter três atributos. Isto pode ser observado quando o conjunto de eventos inseridos no algoritmo C4.5 não possui todos os exemplos necessários para criar uma regra generalizada, neste caso a regra produzida possui atributos a mais para poder cobrir somente os exemplos do conjunto utilizado em sua geração.

A criação de regras sobre especializadas não causa problemas no sistema, entretanto a cobertura da regra é menor do que a regra desejada. No sistema atual, para que a regra sobre especializada amplie sua cobertura, é necessário que uma nova regra seja criada, para complementar a regra existente.

Torna-se importante estudar uma maneira de diminuir a quantidade de sobre especialização das regras, isto será realizados em trabalhos futuros.

6 CONCLUSÃO E TRABALHOS FUTUROS

6.1 Considerações Finais

O tema automação residencial tem sido bastante estudado, as possibilidades ainda não exploradas permitirão o desenvolvimento de muitos trabalhos em diversas áreas, tais como Engenharia, Ciência da Computação, Sociologia, entre outras.

O trabalho desenvolvido propôs um sistema cujo objetivo é observar e aprender regras em uma casa de acordo com o comportamento de seus habitantes, utilizando o conceito de aprendizado com regras de indução.

O sistema proposto partiu da idéia de se armazenar os eventos ocorridos nos atuadores de uma residência, transformá-los em regras e passar a executar estas regras quando as condições dos eventos se repetissem; fazendo uso de técnicas de aprendizado. Somado ao aprendizado foi feito uso de um algoritmo que permitisse não ter de armazenar todas as regras, mas sim generalizar as regras e assim economizar processamento e memória, tal algoritmo é o C4.5, que constrói árvores de decisão.

O sistema proposto tinha como desafios iniciais resolver o problema das regras que eram imediatamente ativadas, causando desconforto ao habitante, e o problema da não detecção de seqüências causais de eventos no tempo, a qual causa armazenamento indevido de eventos. Os problemas foram resolvidos através da criação das Regras Embrionárias e da janela de observação de eventos.

Com base nas soluções identificadas, foi criado um simulador que refletia um dormitório de uma residência.

A lógica do sistema possui algumas variáveis que afetam diretamente o funcionamento do sistema, por tal motivo foi necessário avaliar o comportamento destas variáveis. Para isto foi construído um segundo simulador que permitiu analisar as variáveis.

Os experimentos realizados nos simuladores demonstraram o seguinte, durante este trabalho:

- No primeiro simulador tem-se a indicação de que o sistema funciona, tal como foi proposto, dando destaque para as Regras Embrionárias e a janela de observação de eventos, que funcionaram como foram idealizadas, ou seja, as Regras Embrionárias atuam como uma etapa de validação das regras e a janela

de observação de eventos não permite que seqüências causais de eventos sejam armazenadas para gerarem futuras regras.

- No segundo simulador tem-se a indicação de que é possível manipular as variáveis do sistema até encontrar a melhor configuração das mesmas, de tal forma que o sistema se adaptou ao perfil do usuário e ao ambiente que está sendo automatizado.

Um dos itens avaliados foi a janela de observação, que teve seu correto funcionamento comprovado, entretanto não foram realizados testes para se verificar quais os melhores valores para a janela de observação anterior e para a janela de observação posterior. A avaliação de eventos descartados pela janela de observação deve ser feita com uma residência real e não com um simulador. As simulações com o Banco de Dados de eventos criado não permitiriam avaliar as condições em que a janela de observação deve atuar. Tendo um Banco de Dados colhido de uma residência real não é necessário realizar simulações com o sistema ABC+, basta aplicar a lógica da janela de observação sobre o Banco de Dados e identificar quantos eventos são admitidos e quantos são descartados, perante os valores que são configurados na janela de observação. Na verdade, uma vez tendo colhidos os dados de uma residência é possível calcular quais os valores devem ser utilizados na janela de observação, sem para isto realizar testes.

Como principais pontos positivos identificados na realização deste trabalho destacam-se:

- Tem-se um indicativo de que é possível ter um sistema de Domótica Inteligente que aprende regras em uma residência. Este sistema torna a automação residencial mais acessível a usuários com pouca familiaridade em programar e configurar sistemas, evitando que os mesmos tenham de programar as regras que desejam ter em sua residência.
- Tem-se um indicativo de que é possível configurar o sistema de Domótica Inteligente para se ajustar ao perfil do habitante. As variáveis do sistema permitem criar e eliminar regras com diferentes velocidades e quantidades, ou seja, mais ou menos rapidamente as regras são criadas e/ou eliminadas e mais ou menos regras são criadas e/ou eliminadas.
- Tem-se um indicativo de que é possível disponibilizar novas regras ao habitante de uma residência, sem causar oscilação no banco de regras. As Regras Embrionárias e o desenvolvimento e manutenção das regras permitem

que mesmo no momento em que o habitante está se adaptando a novas regras, as mesmas não sejam descartadas.

- Tem-se um indicativo de que é possível filtrar eventos que não refletem a rotina de um habitante. Eventos que gerariam regras incorretas, pois utilizariam dados de sensores que estão desatualizados, são eliminados.

Não foram realizadas análises com variações de perfis de Banco de Dados de eventos (diferentes agentes) e com variações no número de sensores, atuadores e valores possíveis dos sensores e atuadores. Para que o sistema possa ser estendido para aplicações reais é necessário que sejam feitas novas análises envolvendo tais variações.

Como principais pontos negativos encontrados, durante a realização este trabalho, destacam-se:

- O sistema trabalha atualmente com somente um habitante na residência. Não é feita a diferenciação dos habitantes, se existirem vários habitantes, as regras serão as mesmas para todos.
- Não foi criado um Banco de Dados de eventos que fosse colhido de uma residência real. O Banco de Dados utilizado é hipotético.
- Não foram feitos vínculos entre sensores e atuadores, portanto não foi percebido *loop* entre as regras. A presença de *loop* entre as Regras Ativas pode ser um grande problema.
- Foram encontradas regras com sobre especialização (*overfitting*).

Os pontos negativos podem ser trabalhados de maneira a serem minimizados ou corrigidos.

6.2 Trabalhos Futuros

Para resolver ou aperfeiçoar algumas dos pontos negativos e assim obter uma melhora nos resultados do trabalho realizado, propõe-se alguns trabalhos futuros, descritos a seguir.

6.2.1 Utilização da Tecnologia RFID

Utilização da tecnologia RFID para distinção de habitantes na residência. Alguns sensores podem identificar o usuário através de etiquetas RFID e assim criar bancos de regras específicos para cada habitante.

Como dito anteriormente, o sistema descrito trabalha com um habitante na casa, ou melhor, considera que todos os habitantes se utilizam as mesmas regras. Isto não reflete a realidade em uma casa, pois as pessoas têm diferentes hábitos.

Outra limitação está na situação em que um habitante da casa se aproxima da TV da sala e a liga. Como aprender esta ação?

Para estes casos uma possível solução é o uso da tecnologia RFID (*Radio Frequency Identification*).

Os sistemas RFID são sistemas automáticos de identificação que, usando sinais de rádio freqüência, propiciam a identificação e localização automática de etiquetas eletrônicas portadoras de dados dos itens aos quais as mesmas estão relacionadas.

Um habitante que estiver com seu pijama e este contiver uma etiqueta RFID, poderá se aproximar da TV e ao ligá-la todas estas informações serão armazenadas para gerar futuras regras.

Do mesmo modo as roupas e pertences dos habitantes, contendo etiquetas RFID, poderão servir de identificador de cada habitante. O grande desafio é identificar os habitantes sem cadastramento de suas etiquetas RFID, ou seja, o sistema tem de ser bastante inteligente para associar os itens a cada habitante.

6.2.2 Realização de Testes em uma Residência Real

Um trabalho importante a ser desenvolvido é a colocação de sensores e atuadores em uma residência real. Sendo possível obter dados reais da rotina de usuários. Com um teste prático será possível encontrar as configurações corretas das variáveis do sistema a serem utilizadas em um sistema que possa ser comercializado, ademais será possível também identificar se realmente o sistema está adequado para uso comercial.

6.2.3 Variação no Número de Sensores e Atuadores

Trabalhar com variação no número de sensores e atuadores, bem como com a quantidade de estados destes sensores e atuadores é outro ponto interessante a ser estudado. Com isto será possível detalhar os vínculos existentes nas variáveis do sistema com as variações apontadas. A variação sugerida visa avaliar se existe mudança no comportamento das variáveis, por exemplo, a variável NuEv pode apresentar um gráfico diferente quando

submetida a 6 sensores e a 10 sensores, ou ainda quando os sensores tem 2 estados ou 5 estados.

6.2.4 Simulação e Soluções para *Loop*

Simulação de *loop* entre as regras e soluções para eliminação destes *loops*. Se algum sensor tiver seus estados vinculados aos estados de um atuador, ou de outro sensor, estamos diante de dependências que podem criar um *loop* ou realimentação, que nada mais é do que a execução de uma regra levar a execução de uma segunda regra e outras execuções de regras, até que a primeira regra do *loop* volte a ser acionada, o que não teria mais fim. O *loop* pode levar ao travamento do sistema e causar ainda problemas nos dispositivos da residência. Identificar quando eles aparecem e como eliminá-los, torna-se um trabalho importante.

6.2.5 Minimização de Overfitting

Identificação e execução de soluções que permitam eliminar ou diminuir a super especialização (overfitting) das regras. Tal trabalho permitirá fazer com que o sistema crie as regras desejadas mais rapidamente e com maior precisão.

REFERÊNCIAS BIBLIOGRÁFICAS

AHA, D.W; KIBLER, D.; ALBERT, M. **Instanced-based learning algorithms**. Machine Learning 6, 37-66. 1991.

ANGEL, P. M. **Introducción a la domótica**; Domótica: controle e automação. Escuela Brasileño-Argentina de Informática. EBAI. 1993.

BATISTA, G.E. Pré**-processamento de Dados em Aprendizado de Máquina Supervisionado**. São Carlos, Tese (doutorado), Universidade de São Paulo. 2003.

BOLZANI, C.A.M. **Desenvolvimento de um simulador de controle de dispositivos residenciais inteligentes: uma introdução aos sistemas domóticos**. São Paulo, Dissertação (mestrado), Universidade de São Paulo. 2004a.

BOLZANI, C.A.M. **Residências Inteligentes**. Editora Livraria da Física, São Paulo. 2004b.

BREIMAN, L.;FREDMAN, J.H.;OLSHEN, R.A.; STONE, C.J. **Classification and Regression Trees**. Morgan Kaufmann Publishers, Wadsworth, CA. 1984.

BRETERNITZ, J. V. **Domótica: as casas inteligentes**. Disponível em 20/05/2006 no sítio http://www.widebiz.com.br/gente/vivaldo/domotica.html. 2001.

BROOKS, A.R. **The intelligent Room Project**. In: 2th International Cognitive Technology Conference (ICT'97). Proceedings. Aizu, Japão. 1997.

CAVALHIERI, M.A. **Modelo Comportamental baseado em crenças a teoria Bayesiana para simulações de vida artificial com humanos virtuais**. São Paulo, Dissertação (mestrado), Universidade de São Paulo. 2006.

DE JONG, K. **Learning with Genetic Algorithms: An Overview**. Machine Learning, N° 3, p.121-138. 1988.

DOMINGUES, A.L.S. **Avaliação de critérios e Ferramentas de Teste para Programa 00**. São Carlos, Dissertação (mestrado), Universidade de São Paulo. 2002.

ENG, K. et al. Ada: **Constructing a synthetic organism**. In IEEE/RSJ International Conference on Intelligent Robots and System (IROS 2002). Proceedings. Lausanne. Suiça.

2002.

HOLLAND, J.H. **Escaping brittleness: The possibilities of general-purpose learning algorithms applied to parallel rule-based systems**. Machine Learning: An Artificial Intelligence Approach, p.27. 1986.

HOPFIELD, J.J. **Neural networks and physical systems with emergent collective computational abilities**. In International Academy of Science of the U.S.A., volume 81, p.3088-3092. 1982.

HUNT, E.B.; MARIN, J.; STONE, P.J. **Experiments in Induction**. New York: Academic Press, EE.UU. 1966.

KOBSA, A. **Supporting User Interfaces for All Through User Modeling**. Proceedings of the HCI International, Yokohama, Japão, p. 155-157, 1995.

LOH, W.Y.; SHIH, Y.S. **Split selection methods for classification trees**. Statistica Sinica, vol. 7, pp. 815-840. 1997.

MCCULLOCH, W.S.; PITTS, W. **A logical calculus of the ideas imminent in nervous activity**. In Bulletin of Mathematical Biophysics, volume 5, p.115-133. 1943.

MICHALSKI, R.S.; BRATKO, I.; KUBAT M. **Machine Learning and Data Mining**. Methods and Applications. Wiley & Sons Ltd., EE.UU. 1998.

MINSKY, M.L.; PAPERT, S. **Perceptrons**. MIT Press, Cambridge. 1969.

MITCHELL, T. **Machine Learning**. McGraw Hill. 1997.

MONIZ, L.M.F.F. **SSAA: Sistema Para Simulação de Agentes e Ambientes**. Dissertação (mestrado). Universidade Técnica de Lisboa. Portugal. 1993.

MORGAN, J.; MESSEGER, R. Thaid: **A sequential search program for the analysis of nominal scale dependent variables**. Technical report, Institute for Social Research, University of Michigan. Technical Report. 1973.

MURATORI, J.R. **As tendências do mercado de Automação Residencial**. São Paulo, Congresso Habitar - Congresso de Automação Residencial e Tecnologias para Habitação. 2005.

OSÓRIO, F. S. **Tutorial: Redes Neurais – Aprendizado Artificial**. Disponível em 20/08/2005 no sítio http://www.inf.unisinos.br/~osorio/IForumIA/fia99.pdf. 1999.

QUINLAN, J.R. **Induction of decision trees**. Machine Learning 1, p.81-106. 1986.

QUINLAN, J.R. **C4.5: Programs for Machine Learning**. San Mateo, California, EE.UU, Morgan Kaufmann Publishers. 1993.

QUINLAN, J.R. **MDL and Categorical Theories**. Basser Departament of Computer Science, University of Science, Australia. 1995.

REZENDE, S.O. **Sistemas Inteligentes. Fundamentos e Aplicações**. Editora Manole, São Paulo. 2002.

ROSENBLATT, F. **The Perceptron: A probabilistic Model for Information Storage and Organization in the Brain**. Psychological Review, N°65, p.386-408. 1958.

RUSSELL, S.J.; NORVIG, P. **Inteligência Artificial: tradução da segunda edição**. Elsevier, Rio de Janeiro. 2004.

RUTISHAUSER, U; SCHÄFER, A. **Adaptative Building Automation: A multi-agent approach**. Research project. University of Applied Science Rapperswil. Suiça. 2002a.

RUTISHAUSER, U; SCHÄFER, A. **Adaptative Building Intelligence: A multi-agent approach**. Diploma Thesis. University of Applied Science Rapperswil. Suiça. 2002b.

SANCHES, M. K; GEROMINI, M. R. **Aprendizado de Máquina**. Disponível em 10/07/2005 no sítio http://labic.icmc.usp.br/didatico/am/AM-2001.pdf. 2001.

SANTOS, C. T. **Um Ambiente Virtual Inteligente e Adaptativo baseado em Modelos de Usuário e Conteúdo**. São Leopoldo-RS, Dissertação (mestrado), Universidade do Vale do Rio dos Sinos – Unisinos . 2004.

SERVENTE, M. Algoritmos **TDIDT aplicados a la mineria de datos inteligente**. Buenos Aires, Dissertação (mestrado), Universidad de Buenos Aires. 2002.

SIERRA, E.A. et al. **Sistema experto para control inteligente de las variables ambientales de un edificio energéticamente eficiente**. XI Reunión de Trabajo en Procesamiento de la Información y Control. Buenos Aires, Argentina. 2005.

SIMON, H.A. **Search and Reasoning in Problem Solving**. Artificial Intelligence, N°21, p.7-30. 1983.

TONIDANDEL, F. **Desenvolvimento e implementação de um sistema de planejamento baseado em casos**. São Paulo, Tese (doutorado), Universidade de São Paulo. 2003.

TONIDANDEL, F; TAKIUCHI, M.; MELO, E. **Domótica Inteligente: Automação baseada em comportamento**. Congresso Brasileiro de Automática. 2004.

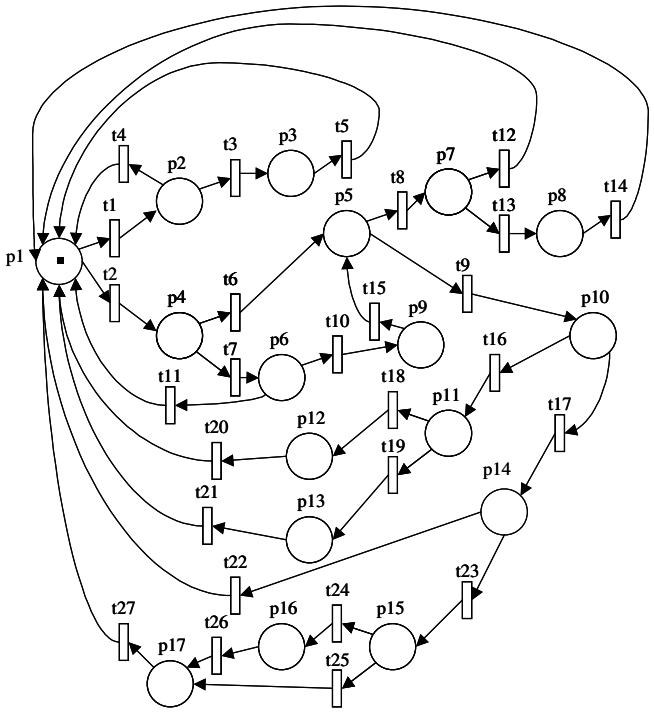
UTGOFF, P. E. **Incremental Induction of Decision Trees**. Machine Learning, vol. 4, Págs. 161-186. 1989.

WOOLDRIDGE, M; RAO, A. **Foundations of rational agency**. Kluwer, Dordrecht, Netherlands. 1999.

APÊNDICE 1

REDE DE PETRI DO SISTEMA ABC+

A seguir é apresentada a Rede de Petri do funcionamento do sistema:



LEGENDA
EA=Evento Atuador
ES=Evento Sensor
BE=Banco de Eventos
RA=Regra Ativa
BRA=Banco de Regras Ativas
RE=Regra Embrionária
BRE=Banco de Regras Embrionárias

POSIÇÕES
P1 – Estado inicial de espera
P2 – Busca por RA existente
P3 – Realização da ação X, fazer ATIV=ATIV+1, com ATIV máximo=T e ordenação do BRA por ATIV
P4 – Busca por ES anterior recente (R segundos)
P5 – Busca por RA existente
P6 – Busca por ES posterior recente (R segundos)
P7 – Avaliação se a regra é igual ou contrária
P8 – Fazendo EXC=EXC+1, se EXC>EXC_At a regra é excluída do BRA
P9 – Armazenamento dos dados do ES posterior
P10 – Busca por RE existente
P11 – Avaliação se a regra é igual ou contrária
P12 – Fazendo OK+OK+1, se OK=OK_Emb regra vira RA
P13 - Fazendo NOK=NOK+1, se NOK>NOK_Emb regra é excluída do BRE
P14 – Colocação do evento no BE
P15 – C4.5 gera novas regras a partir de BE
P16 – As regras repetidas são ignoradas
P17 - Regras antigas do BRE com OK<OK_Emb_exc são excluídas; coloca regras novas no BRE; para toda RA fazer ATIV=ATIV-1; se RA tem ATIV<ATIV_At coloca-a no BRE

TRANSIÇÕES
T1 – Novo evento de sensor
T2 – Novo evento de atuador
T3 – Existência de RA
T4 – Não existência de RA
T5 – Finalização da ação em P3
T6 – Existência de ES anterior
T7 – Não existência de ES anterior
T8 – Existência de RA
T9 – Não existência de RA
T10 - Existência de ES posterior
T11 – Não existência de ES posterior
T12 – A regra é igual
T13 – A regra é contrária
T14 – Finalização da ação em P8
T15 – Finalização da ação em P9
T16 - Existência de RE
T17 – Não existência de RE
T18 - A regra é igual
T19 – A regra é contrária
T20 - Finalização da ação em P12
T21 – Finalização da ação em P13
T22 – BE não têm NuEv eventos
T23 – BE têm NuEv eventos
T24 – Alguma(s) nova(s) regra(s) está(ão) em BRA ou BRE
T25 – Nenhuma nova regra está em BRA ou BRE
T26 - Finalização da ação em P16
T27 – Finalização da ação em P17

Figura A1.1 – Rede de Petri do sistema ABC+.

APÊNDICE 2

TABELAS DAS SIMULAÇÕES REALIZADAS

As tabelas a seguir foram resultados das simulações feitas e serviram para gerar os gráficos apresentados nas análises quantitativas.

Foram utilizadas as seguintes definições:

EC-1, EC-2 e EC-3: Número de Regras Embrionárias criadas utilizando o bando de dados de teste 1, 2 e 3, respectivamente.

EE-1, EE-2 e EE-3: Número de Regras Embrionárias excluídas utilizando o bando de dados de teste 1, 2 e 3, respectivamente.

AC-1, AC-2 e AC-3: Número de Regras Ativas criadas utilizando o bando de dados de teste 1, 2 e 3, respectivamente.

AE-1, AE-2 e AE-3: Número de Regras Ativas excluídas (rebaixadas) utilizando o bando de dados de teste 1, 2 e 3, respectivamente.

Simulações com a Variável NuEv

As simulações foram feitas variando o valor de NuEv desde 20 até 500, mantendo os seguintes valores das outras variáveis:

- OK_Emb = 20;
- NOK_Emb = 7;
- EXC_At = 20;
- ATIV_At = 2;
- OK_Emb_exc = 2.

Tabela A2.1 – Resultados das simulações da variável NuEv.

NuEv	EC-1	EC-2	EC-3	EC	EE-1	EE-2	EE-3	EE	AC-1	AC-2	AC-3	AC	AE-1	AE-2	AE-3	AE
20	95	91	96	94	67	71	74	71	25	16	19	20	23	13	17	18
40	69	72	84	75	48	49	51	49	18	19	25	21	16	16	21	18
60	56	65	74	65	33	49	42	41	22	12	23	19	18	9	20	16
80	67	71	58	65	37	40	36	38	20	12	19	17	17	8	16	14
100	62	49	53	55	43	34	34	37	10	10	15	12	6	6	12	8
150	55	44	44	48	37	29	21	29	11	13	15	13	5	8	9	7
200	48	45	52	48	29	30	25	28	11	13	12	12	7	9	9	8
300	37	30	30	32	19	21	16	19	10	7	11	9	4	1	5	3
500	26	36	32	31	6	15	21	14	7	6	9	7	0	0	1	0

As simulações feitas com os outros três Bancos de Dados onde os eventos esporádicos foram suprimidos possuem exatamente as mesmas variações e valores das variáveis e os resultados foram:

Tabela A2.2 – Resultados das simulações da variável NuEv, com BD somente com regras desejadas.

NuEv	EC-1	EC-2	EC-3	EC	EE-1	EE-2	EE-3	EE	AC-1	AC-2	AC-3	AC	AE-1	AE-2	AE-3	AE
20	16	22	13	17	6	6	4	5	9	14	9	11	6	11	6	8
40	9	14	17	13	1	1	3	2	8	10	9	9	3	7	8	6
60	7	11	11	10	1	0	2	1	6	10	7	8	1	6	6	4
80	12	10	9	10	0	0	0	0	10	8	7	8	6	4	3	4
100	12	6	8	9	0	1	0	0	10	5	7	7	6	0	2	3
150	7	7	10	8	1	0	0	0	6	6	7	6	0	0	2	1
200	8	7	8	8	0	0	0	0	7	6	7	7	0	0	0	0
300	8	7	7	7	0	0	0	0	7	6	6	6	0	0	0	0

Simulações com a Variável OK_Emb

As simulações foram feitas variando o valor de OK_Emb desde 5 até 30, mantendo os seguintes valores das outras variáveis:

- NuEv = 200;
- NOK_Emb = 7;
- EXC_At = 20;
- ATIV_At = 2;
- OK_Emb_exc = 2.

Tabela A2.3 – Resultados das simulações da variável OK_Emb.

OK_Emb	EC-1	EC-2	EC-3	EC	EE-1	EE-2	EE-3	EE	AC-1	AC-2	AC-3	AC	AE-1	AE-2	AE-3	AE
5	65	56	67	63	15	18	19	17	44	37	46	42	39	30	41	37
10	44	53	52	50	23	27	19	23	13	19	30	21	6	15	25	15
15	46	44	58	49	28	28	27	28	14	13	25	17	9	10	22	14
20	48	45	52	48	29	30	25	28	11	13	12	12	7	9	9	8
25	38	48	50	45	26	31	25	27	7	16	12	12	2	11	8	7
30	39	45	51	45	27	31	35	31	7	12	12	10	2	8	8	6

Simulações com a Variável NOK_Emb

As simulações foram feitas variando o valor de NOK_Emb desde 5 até 30, mantendo os seguintes valores das outras variáveis:

- NuEv = 200;
- OK_Emb = 20;
- EXC_At = 20;
- ATIV_At = 2;
- OK_Emb_exc = 2.

Tabela A2.4 – Resultados das simulações da variável NOK_Emb.

NOK_Emb	EC-1	EC-2	EC-3	EC	EE-1	EE-2	EE-3	EE	AC-1	AC-2	AC-3	AC	AE-1	AE-2	AE-3	AE
5	41	47	61	50	26	33	40	33	9	12	10	10	4	7	6	6
10	49	44	44	46	21	22	22	22	13	10	19	14	8	6	15	10
15	35	34	40	36	13	16	17	15	18	15	19	17	12	10	14	12
20	39	32	35	35	10	13	9	11	19	15	21	18	15	10	17	14
25	40	37	34	37	11	11	7	10	16	18	20	18	13	13	15	14
30	37	37	30	35	8	9	7	8	19	17	18	18	14	14	13	14

Simulações com a Variável EXC_At

As simulações foram feitas variando o valor de EXC_At desde 5 até 40, mantendo os seguintes valores das outras variáveis:

- NuEv = 200;
- OK_Emb = 20;
- NOK_Emb = 7;
- ATIV_At = 2;
- OK_Emb_exc = 2.

Tabela A2.5 – Resultados das simulações da variável EXC_At.

EXC_At	EC-1	EC-2	EC-3	EC	EE-1	EE-2	EE-3	EE	AC-1	AC-2	AC-3	AC	AE-1	AE-2	AE-3	AE
5	58	53	66	59	31	29	25	28	20	18	25	21	18	16	23	19
10	44	50	65	53	29	32	27	29	11	16	22	16	9	12	20	14
15	45	49	48	47	30	30	26	29	9	16	14	13	4	13	11	9
20	48	45	52	48	29	30	25	28	11	13	12	12	7	9	9	8
25	41	45	44	43	29	28	24	27	6	12	11	10	1	9	8	6
30	40	45	48	44	30	28	23	27	6	12	13	10	1	9	9	6
35	40	42	40	41	30	27	25	27	6	11	12	10	1	6	7	5
40	40	38	40	39	29	26	24	26	5	11	12	9	1	5	7	4

Simulações com a Variável ATIV_At

As simulações foram feitas variando o valor de ATIV_At desde 1 até 300, mantendo os seguintes valores das outras variáveis:

- NuEv = 200;
- OK_Emb = 5;
- NOK_Emb = 50;
- EXC_At = 500;
- OK_Emb_exc = 1.

Tabela A2.6 – Resultados das simulações da variável ATIV_At.

ATIV_At	EC-1	EC-2	EC-3	EC	EE-1	EE-2	EE-3	EE	AC-1	AC-2	AC-3	AC	AE-1	AE-2	AE-3	AE
1	27	22	14	21	1	1	0	1	22	15	12	16	8	1	1	3
100	28	22	14	21	1	1	0	1	22	15	12	16	9	1	1	4
200	29	23	14	22	1	1	0	1	24	15	12	17	10	3	1	5
300	35	27	14	25	1	1	0	1	24	19	12	18	16	6	1	8

Simulações com a Variável OK_Emb_exc

As simulações foram feitas variando o valor de OK_Emb_exc desde 1 até 17, mantendo os seguintes valores das outras variáveis:

- NuEv = 100;
- OK_Emb = 20;
- NOK_Emb = 20;
- EXC_At = 20;
- ATIV_At = 2.

Tabela A2.7 – Resultados das simulações da variável OK_Emb_exc.

OK_Emb_exc	EC-1	EC-2	EC-3	EC	EE-1	EE-2	EE-3	EE	AC-1	AC-2	AC-3	AC	AE-1	AE-2	AE-3	AE
1	41	39	43	41	15	17	12	15	12	14	25	17	5	10	22	12
5	38	44	43	42	17	22	12	17	12	17	25	18	5	13	22	13
9	52	45	43	47	22	20	12	18	17	17	25	20	13	13	22	16
13	55	49	43	49	33	25	12	23	16	18	25	20	11	14	22	16
17	55	55	47	52	40	24	13	26	11	25	27	21	9	22	25	19