



UNIVERSIDADE ESTADUAL DO PIAUÍ-UESPI  
CAMPUS PROF. ALEXANDRE ALVES DE OLIVEIRA  
BACHARELADO EM CIÊNCIA DA COMPUTAÇÃO

ELLYYNNE CHRISTINE RIBEIRO DE MORAES SOUSA

**DOMÓTICA INTELIGENTE: SIMULADOR DO SISTEMA ABC+ PARA  
APRENDIZADO BASEADO EM COMPORTAMENTO**

Biblioteca UESPI - PHB  
Registro Nº M 406  
CDD 006.3  
CUTTER 5725.d  
V \_\_\_\_\_ EX. 01  
Data 05/04/11  
Visto. [assinatura]

Parnaíba – PI

2011

**ELLAYNNE CHRISTINE RIBEIRO DE MORAES SOUSA**

**DOMÓTICA INTELIGENTE: SIMULADOR DO SISTEMA ABC+ PARA  
APRENDIZADO BASEADO EM COMPORTAMENTO**

Monografia apresentada à Universidade Estadual do Piauí – UESPI, como requisito para obtenção do título de Bacharel em Ciência da Computação.

Orientador: Dario Brito Calçada

Parnaíba

2011

FICHA CATALOGRÁFICA ELABORADA PELO BIBLIOTECÁRIA  
CÁTIA REGINA FURTADO DA COSTA CRB-3/1109

S725a Sousa, Ellayne Christine Ribeiro de Moraes.

Domótica inteligente: simulador do sistema ABC+ para  
aprendizado baseado em comportamento. / Ellayne Christine  
Ribeiro de Moraes Sousa. – Parnaíba, 2011.

69 f.

Monografia de conclusão de curso de Bacharelado em  
Ciências da Computação, Universidade Estadual do Piauí,  
Parnaíba, 2011.

Orientador: Prof. Esp. Dario Brito Calçada.

1. Inteligência artificial – domótica. 2. Sistema ABC+ –  
Automação. 3. Aprendizado automático – Software. I. Título.

CDD – 006.3

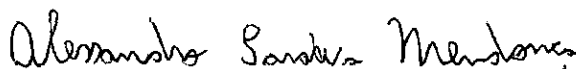


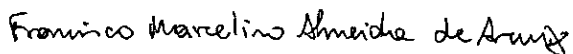
## Ata de Defesa de Trabalho de Conclusão de Curso

Aos dezoito dias do mês de março de dois mil e onze, às 10h30, no Laboratório de Informática do Campus Prof. Alexandre Alves Oliveira - UESPI realizou-se a sessão pública de defesa do Trabalho de Conclusão de Curso da aluna **Ellayne Christine Ribeiro de Moraes Sousa** intitulado **ABC+ implementação de uma ferramenta para domótica inteligente**, como parte dos requisitos necessários para a obtenção do grau de Bacharel em Ciência da Computação. A Banca Examinadora foi presidida pelo Prof. Dario Brito Calçada e composta pelos membros efetivos os professores Alessandro Saraiva Mendonça e Francisco Marcelino Almeida de Araújo. Aberta a sessão pública, a candidata teve a oportunidade de expor seu trabalho. Após a exposição, a aluna foi arguida oralmente pelos membros da Banca Examinadora. Em seguida, em sessão reservada e nos termos do Regulamento Geral dos Trabalhos de Conclusão de Curso da UESPI, os membros da Banca realizaram a avaliação do candidato, que obteve nota 9,6 (nove pontos) sendo, portanto, **aprovada**. Nada mais havendo a tratar, eu, Prof. Dario Brito Calçada, lavrei a presente Ata que será lida e assinada por mim, pelos demais membros da Banca Examinadora e pela candidata. Parnaíba (PI), 19 de março de 2011.

### Banca Examinadora

  
Prof. Esp. Dario Brito Calçada  
Orientador, UESPI

  
Prof. Esp. Alessandro Saraiva Mendonça  
Avaliador, PVP

  
Prof. Esp. Francisco Marcelino Almeida de Araújo  
Avaliador, FPI

### Candidato

  
Ellayne Christine Ribeiro de Moraes Sousa

Dedico este trabalho ao meu pai, minha mãe,  
minha irmã e amigos, pessoas especiais que  
agradeço por ter em minha vida.

## AGRADECIMENTOS

- Primeiramente, agradeço às forças superiores por me iluminar e dar forças para conquistar mais esta vitória na minha vida;
- A minha mãe Cristina, meu pai Batista, minha irmã Layanne e aos demais familiares pelo amor, compreensão, paciência e por acreditarem no meu potencial e estarem sempre ao meu lado;
- Ao Gibb, Kitty, Fadinha, Pingo, que já se foram e à Pucca e ao Valente por darem tanta alegria a minha vida;
- Aos meus amigos da UESPI (companheiros de turma e professores) pelos bons momentos de alegria e por estarmos sempre unidos nestes quatro anos;
- Aos demais amigos, amigos verdadeiros que sempre estiveram comigo, me apoiando, incentivando e demonstrando suas amizades;
- Ao professor Dario por me acompanhar desde a escola até a universidade;
- Enfim, a todos que acreditaram e torceram por mim. Muito obrigada.

"Imaginação é mais importante que conhecimento."

Albert Einstein

## RESUMO

A área de automação residencial ou domótica tem recebido uma maior atenção ultimamente. Mas a maioria das aplicações ainda envolve uma arquitetura centralizada e funções pré-programadas das ações a serem executadas na casa. Em vista ao crescente avanço das tecnologias e da Inteligência Artificial já é possível falar de Domótica Inteligente. Neste trabalho será estudado o sistema ABC+ (Automação Baseada em Comportamento) que cria regras a partir da observação do comportamento de um habitante em uma casa.

O sistema em questão utiliza conceitos de aprendizado automático por indução, encontrados na Inteligência Artificial, para a criação e manutenção das regras. Através de uma avaliação nas sequências de mudanças de estado em sensores e atuadores é possível extrair informações dos costumes do habitante de forma transparente. As regras são criadas pelo próprio sistema sem necessidade de nenhuma intervenção do usuário. Um simulador incorporando a lógica de funcionamento do ABC+ foi desenvolvido para demonstrar como é realizada a aquisição de conhecimento para alimentar o sistema domótico inteligente.

**Palavras-chave:** Domótica, Inteligência Artificial, Aprendizado Automático.



## **ABSTRACT**

The field of home automation or domotics has received an increased attention lately. But most of the applications still involve a centralized architecture and preprogrammed functions of the actions to be performed at home. Given the increasing advancement of technology and artificial intelligence it is already possible to speak of Intelligent Domotics. This work will study the ABC system + (Behavior-Based Automation) establishing rules from observing the behavior of an inhabitant in a house. The system in question uses the concepts of automatic learning by induction, found in Artificial Intelligence, for the creation and maintenance of rules. Through an evaluation in the sequences of state changes in sensors and actuators it's possible to extract information from the customs of the inhabitants of a transparent manner. Rules are created by the system itself without any user intervention. A simulator incorporating the operating logic of the ABC + was developed to show how is done the acquisition of knowledge to feed the smart home automation system.

**Key-words:** Domotics, Artificial Intelligence, Automatic Learning.

## LISTA DE FIGURAS

<b>Figura 1:</b> <i>Esquema da fase de Treinamento no processo de Aprendizado Automático..</i>	19
<b>Figura 2:</b> <i>Passos que compõem o processo de KDD.....</i>	32
<b>Figura 3:</b> <i>Exemplo de uma árvore de decisão.....</i>	33
<b>Figura 4:</b> <i>Árvore formada para o conjunto de observações da Tabela 1 .....</i>	35
<b>Figura 5:</b> <i>Pseudocódigo do algoritmo C4.5 .....</i>	36
<b>Figura 6:</b> <i>Regras geradas pelo C4.5 para o problema Jogar Golf.....</i>	37
<b>Figura 7:</b> <i>Arquiteturas dos sistemas ABC e ABC+ .....</i>	43
<b>Figura 8:</b> <i>Janela de observação do Sistema ABC+.....</i>	45
<b>Figura 9:</b> <i>Fluxograma da lógica da Janela de Observação.....</i>	46
<b>Figura 10:</b> <i>Fluxograma da lógica de validação da Regra Embrionária .....</i>	47
<b>Figura 11</b> <i>Esquema da lógica de Manutenção das Regras .....</i>	50
<b>Figura 12:</b> <i>Fluxograma do funcionamento do Sistema ABC+ .....</i>	52
<b>Figura 13:</b> <i>Tela principal do simulador do Sistema ABC+.....</i>	56
<b>Figura 14:</b> <i>Execução da Janela de Observação.....</i>	57
<b>Figura 15:</b> <i>Janela para inserção das Regras Embrionárias .....</i>	58
<b>Figura 16:</b> <i>Janela de Manutenção das Regras Embrionárias e Ativas .....</i>	59

## LISTA DE TABELAS

<i>Tabela 1: Condições para se jogar ou não Golf.....</i>	34
<i>Tabela 2: Exemplo de um banco de Eventos.....</i>	48
<i>Tabela 3: Exemplo de um banco de Regras Ativas.....</i>	48
<i>Tabela 4: Exemplo de um banco de Regras Embrionárias.....</i>	48
<i>Tabela 5: Tabela de Eventos.....</i>	55
<i>Tabela 6: Tabela de Regras Embrionárias.....</i>	55
<i>Tabela 7: Tabela de Regras Ativas.....</i>	55

## SUMÁRIO

<b>1 INTRODUÇÃO</b> .....	<b>13</b>
<b>2 INTELIGÊNCIA ARTIFICIAL</b> .....	<b>15</b>
2.1 Introdução a Inteligência Artificial.....	15
2.2 Aprendizado de Máquina ou Automático.....	19
2.3 Formas de Aprendizado.....	20
2.3.1 <i>Supervisionado</i> .....	21
2.3.2 <i>Não Supervisionado</i> .....	21
2.3.3 <i>Semi-supervisionado ou por reforço</i> .....	21
2.4 Paradigmas de Aprendizado Automático.....	22
2.4.1 <i>Paradigma Simbólico</i> .....	22
2.4.2 <i>Paradigma Estatístico</i> .....	22
2.4.3 <i>Paradigma Instance-based</i> .....	23
2.4.4 <i>Paradigma Conexionista</i> .....	23
2.4.5 <i>Paradigma Genético</i> .....	24
2.4.6 <i>Paradigma Baseado em Exemplos</i> .....	24
2.5 Estratégias de Aprendizado Automático.....	25
2.5.1 <i>Aprendizado por Hábito</i> .....	25
2.5.2 <i>Aprendizado por Instrução</i> .....	25
2.5.3 <i>Aprendizado por Dedução</i> .....	25
2.5.4 <i>Aprendizado por Analogia</i> .....	26
2.5.5 <i>Aprendizado por Indução</i> .....	26
<b>3 MINERAÇÃO DE DADOS</b> .....	<b>28</b>
3.1 Descoberta de Conhecimento em Banco de Dados.....	28
3.2 Pré-processamento.....	28
3.3 Mineração de Dados.....	29
3.3.1 <i>Associação</i> .....	29
3.3.2 <i>Classificação</i> .....	30
3.3.3 <i>Agrupamento</i> .....	30
3.3.4 <i>Regressão</i> .....	31
3.4 Pós-processamento.....	31
3.5 Indução de Árvores de Decisão.....	32
3.6 O Algoritmo C4.5.....	35
3.7 See5/C5.0.....	38
<b>4 DOMÓTICA</b> .....	<b>39</b>
4.1 História da Domótica.....	40
4.2 Domótica Inteligente.....	40
4.3 Sistema ABC+.....	41
4.3.1 <i>Janela de Observação</i> .....	44
4.3.2 <i>Regras Embrionárias</i> .....	46
4.3.3 <i>Desenvolvimento e Manutenção das Regras</i> .....	47
4.3.4 <i>Inconsistência nas Regras</i> .....	53
<b>5 IMPLEMENTAÇÃO DO SISTEMA ABC+.....</b>	<b>54</b>

5.1 Observação e Armazenamento de Novos Eventos .....	56
5.2 Inserção das Regras Embrionárias.....	58
5.3 Manutenção das Regras Ativas e Embrionárias .....	59
<b>6 CONCLUSÕES E TRABALHOS FUTUROS .....</b>	<b>62</b>
6.1 Considerações Finais .....	62
6.2 Trabalhos Futuros .....	62
<b>REFERÊNCIAS BIBLIOGRÁFICAS .....</b>	<b>64</b>
<b>ANEXO A: EVENTOS DE SENSOR E ATUADOR .....</b>	<b>68</b>
<b>ANEXO B: REDE DE PETRI DO SISTEMA ABC+ .....</b>	<b>69</b>

## 1 INTRODUÇÃO

A popularização dos computadores pessoais, da Internet, dos dispositivos móveis, dentre outras invenções fez com que a maioria das pessoas se familiarizasse com equipamentos que integram essas tecnologias. É natural que se espere uma evolução em outras áreas, como nas residências e suas rotinas domésticas (BOLZANI, 2010).

Domótica é a junção das palavras *domus* (que significa casa) e robótica (controle automatizado de algo). Ela pode ser definida como o controle automático dos recursos de uma residência, como o controle da iluminação; aquecimento e ventilação, irrigação, temperatura da água, entre outros, através da instalação de tecnologias, tornando-as o que popularmente se conhece por “casas inteligentes”. Uma casa inteligente combina funções de segurança, economia, conforto e comodidade, ecologia, integração e entretenimento.

A diferença essencial entre um edifício inteligente e um edifício que utiliza de tecnologias tradicionais está na forma como todas as funcionalidades se integram e se complementam, compartilhando a informação entre os diferentes sistemas (de segurança, de climatização, de eletrodomésticos, etc) que possui (ALVES; MOTA, 2003).

Com o avanço das tecnologias e do conhecimento, principalmente em Inteligência Artificial, pode se alcançar o verdadeiro sentido do termo “Casas Inteligentes” que é um ambiente automatizado agindo de forma a atender as necessidades e preferências de seus moradores, atendendo aos requisitos já citados como integração, segurança, economia e conforto.

Surgiu assim o conceito de Domótica Inteligente (DI), que muito se confunde com o conceito de automação (ou simplesmente domótica). A Domótica Inteligente se distingue da Domótica simples por incorporar algum mecanismo automático de tomada de decisão baseada em técnicas de Inteligência Artificial. A arquitetura da Domótica Inteligente deve ser descentralizada e adaptativa às necessidades dos habitantes e às resoluções de problemas e gerenciamento de recursos.

O comportamento dos seres humanos muda ao longo do tempo, bem como cada indivíduo possui suas preferências. A DI propõe uma inversão de responsabilidades no gerenciamento de uma casa automatizada, que ao contrário dos habitantes terem de se adaptar ao funcionamento pré-programado desta casa, o sistema da DI é que deve adaptar suas regras de automação ao comportamento dos seus habitantes (TONIDADEL; TAKIUCHI; MELO, 2004).

Neste trabalho é apresentada uma definição e implementação de um sistema domótico inteligente, o ABC+ (Automação Baseada em Comportamento), proposto por Sgarbi e Tonidandel (2006a). O ABC+ usa técnicas de aprendizado no ajuste das regras de automação ao comportamento dos habitantes de uma casa, que se dá de forma automática e sem intervenção do usuário, sem que haja uma configuração manual.

Além do sistema de domótica mencionado são abordados conceitos relacionados à Inteligência Artificial e Mineração de Dados que são fundamentais para aprofundamento do assunto.

## 2 INTELIGÊNCIA ARTIFICIAL

### 2.1 Introdução à Inteligência Artificial

Com a seguinte frase: “Eu sugiro considerar a questão: As máquinas podem pensar? Isto deve começar com as definições dos significados dos termos ‘máquina’ e ‘pensar’”, Turing (1950) em seu artigo *Computing Machinery and Intelligence* levantou questões importantes para o desenvolvimento da Inteligência Artificial que vai desde o estudo do funcionamento do cérebro humano até a possibilidade de ‘transferir’ essa habilidade para as máquinas.

Assim, antes de adentrar ao assunto de Inteligência Artificial (IA) é importante que se faça uma breve explanação sobre dois conceitos fundamentais para entender em que se baseiam as definições encontradas para a IA: O que é Inteligência e o que é Aprendizado.

Larousse (1999) define inteligente como sendo o ser dotado de inteligência, capaz de compreender, esperto, habilidoso. E ainda como a faculdade de conhecer, aprender, conceber e compreender, onde essa inteligência distingue o ser humano dos outros animais. Mas essas definições apresentam contradições (OSÓRIO, 1999) já que os animais são capazes de conhecer (distinguir seus donos de desconhecidos e o caminho onde está sua água e comida, por exemplo), de compreender (alguma instrução, como sentar, deitar, etc) e ter habilidades.

Aplicando um questionário a diferentes pessoas sobre o que elas definiriam como Inteligência se tem respostas como: capacidade de aprender algo, conhecimento a respeito de determinada ou diversificadas áreas, capacidade de escolha, e assim por diante. No dicionário Priberam (2010), encontramos as seguintes explicações:

1. Conjunto de todas as faculdades intelectuais (memória, imaginação, juízo, raciocínio, abstração e concepção).
2. Qualidade de inteligente.
3. Compreensão fácil.
4. Pessoa muito inteligente e erudita.
5. Fig. Acordo, conluio.
6. Harmonia.
7. Habilidade.

Conclui-se, então que há muitas abordagens para definir o que seja Inteligência, cada uma focando aquilo que mais se destaca no ponto de vista e área de estudo de seu autor. Se pode afirmar ainda que a Inteligência seja formada por um conjunto de fatores que a caracterizam e a tornam uma área de estudo ampla e complexa.

O aprendizado é outro conceito importante a ser argumentado, pois este é um ponto de destaque dentro da IA. Não há como falar de inteligência sem citar aprendizado, pois uma das



características já apontadas para definir o termo Inteligente é a capacidade de adquirir conhecimentos.

Para Osório (1999), aprendizado é a capacidade de se adaptar, de modificar e melhorar seu comportamento e suas respostas sendo, portanto, uma das prioridades mais importantes dos seres considerados inteligentes sejam eles humanos ou não. Assim, dentro da definição dada, este autor cita itens diretamente ligados ao aprendizado:

- Adaptação e mudança de comportamento de forma a evoluir, melhorar algum ponto que seja tido como insatisfatório para alguma exigência;
- Correção dos erros cometidos, de forma a não realizá-los mais no futuro;
- Otimização, onde através de melhorias nas ações realizadas impliquem em alguma melhora, como, por exemplo, o gasto para se realizar uma tarefa ou a redução do tempo que se leva para completar uma tarefa;
- Interação com o meio/pessoas para que seja possível a troca de experiências, onde se podem adquirir novos conhecimentos como aprender com os erros cometidos por outros;
- Representação do conhecimento adquirido. O sistema deve armazenar uma grande quantidade de informações e isto requer uma forma ótima de representar esse conhecimento que permita ao sistema explorá-los de maneira conveniente, já que há uma limitação de recursos físicos.

Com as definições apresentadas a respeito dos termos Inteligência e Aprendizado, podemos partir para as definições de Inteligência Artificial e Aprendizado Automático.

O primeiro trabalho reconhecido como IA foi realizado em 1943, quando Warren McCulloch e Walter Pitts se basearam no conhecimento da fisiologia básica e da função dos neurônios no cérebro, em uma análise formal da lógica proposicional criada por Russel e Whitehead e na teoria da computação de Turing para propor um modelo de neurônios artificiais. Esses dois pesquisadores também sugeriram que tais redes de neurônios artificiais se definidas adequadamente seriam capazes de aprender.

Em 1951, Marvin Minsky e Dean Edmonds, do Departamento de Matemática de Princeton, construíram o primeiro computador de rede neural, o SNARC. Mais tarde, Minsky acabou provando teoremas importantes sobre as limitações da pesquisa em redes neurais.

Alan Turing foi quem primeiro idealizou uma visão completa da IA em seu artigo “*Computing Machinery and Intelligence*”, publicado em 1950, onde apresentou o Teste de Turing, o aprendizado de máquina, algoritmos genéticos e aprendizado por reforço (RUSSEL; NORVIG, 2004).

John McCarthy, do Dartmouth College em Princeton, convenceu Minsky, Claude Shannon e Nathaniel Rochester a reunir pesquisadores interessados em teoria dos autômatos, redes neurais e estudo da inteligência artificial. Eles participariam de um seminário de dois meses durante o ano de 1956. No total, havia 10 participantes, incluindo Trechard More de Princeton, Arthur Samuel da IBM e Ray Solomonoff e Oliver Selfridge do MIT. Neste mesmo seminário, McCarthy sugeriu o nome para o campo que viria a crescer nos anos seguintes: Inteligência Artificial.

Na história da IA houve momentos de sucesso, otimismo impróprio e quedas resultantes do entusiasmo de seus pesquisadores, assim como houve ciclos de introdução de novas abordagens criativas e de aprimoramento sistemáticos das melhores estratégias. O avanço da IA se deu mais rapidamente nas últimas décadas devido à adoção do método científico nas experiências e comparações entre as abordagens (RUSSEL; NORVIG, 2004).

Assim como há várias definições para descrever o que seja Inteligência, conseqüentemente há várias definições para IA também. Alguns conceitos que podemos encontrar para IA são:

- Conjunto de teorias e técnicas empregadas com a finalidade de desenvolver máquinas capazes de simular a inteligência humana (LAROUSSE, 1999).
- Área de estudos da computação que se interessa pelo estudo e criação de sistemas que possam exibir um comportamento inteligente e realizar tarefas complexas com um nível de competência que é equivalente ou superior ao de um especialista humano (NIKOLOPOULOS, 1997).
- O estudo das idéias que permitem aos computadores serem inteligentes (WINSTON, 1984).

A ideia de se implantar Inteligência em máquinas para que estas possam ser capazes de realizar atividades ditas inteligentes envolve vários pontos que vêm sendo propostos. Russel e Norvig (2004) apontam algumas características que devem ser aplicadas aos sistemas de IA para que estes sejam capazes de executar as funções esperadas:

- Processamento de linguagem natural para que se haja uma comunicação em idioma natural.
- Representação do conhecimento para armazenar as informações cedidas e o que foi aprendido.
- Raciocínio para processar as informações armazenadas de forma a responder a novas questões e tomada de decisões.
- Aprendizado para se adaptar a novas circunstâncias e identificar e usar padrões.

Estes autores citam ainda ciências antigas às quais a IA se fundamenta e que contribuíram e contribuem com ideias e técnicas para ajudar a compreender melhor o seu objetivo. São elas: Filosofia, Matemática, Economia, Neurociência, Psicologia, Engenharia de Computadores, Teoria de Controle e Cibernética e a Linguística.

A Filosofia (surgiu em 428 a.C.), com suas teorias de raciocínio e aprendizado, comparou o cérebro a uma máquina, que trabalha sob um conhecimento codificado em uma linguagem interna. A Matemática (surgiu por volta de 800 d.C.) oferece as teorias formais de lógica, probabilidade, tomada de decisões e a computação/cálculo, além de definir a base para a compreensão do raciocínio sobre algoritmos.

Os economistas (a Economia surgiu em 1776) formalizaram o problema de tomar decisões que maximizam o resultado esperado na tomada de decisões. Para investigar a mente humana, a Psicologia (1879) e a Neurociência realizam estudos do sistema nervoso, como o cérebro habilita o pensamento e como os seres humanos e animais processam as informações (RUSSEL; NORVIG, 2004).

Para tornar possíveis as aplicações de IA, a Engenharia de Computadores (1940) forneceu os artefatos necessários para o funcionamento dos programas de forma cada vez mais eficiente. Com a contribuição da Teoria do Controle e da Cibernética a IA lida com o projeto de dispositivos que agem de forma ótima com base no *feedback* do ambiente.

A Linguística dispõe as teorias da estrutura e significado da linguagem. Finalmente, da Ciência da Computação temos as ferramentas com as quais podemos tornar a Inteligência Artificial uma realidade.

Para complementar as ideias anteriores, a afirmação abaixo resume de que forma se pode iniciar um estudo de IA levando-se em conta que características da Inteligência se desejam transferir a um computador:

Se quisermos reproduzir a inteligência humana, devemos ter consciência de quais aspectos da inteligência que gostaríamos que fossem reproduzidos, e que em nossa opinião, permitiriam o desenvolvimento de um sistema com propriedades ditas “inteligentes” (OSÓRIO, 1999).

## 2.2 Aprendizado de Máquina ou Automático

O aprendizado automático ou de máquina (AM) é uma área da IA que busca métodos computacionais ligados a aquisição de conhecimento (SGARBI, 2007) e constrói sistemas que possuem capacidade de aprender através de experiências, além de estudar formas para organizar o conhecimento já existente (MITCHELL, 1997). Esses sistemas devem ter a habilidade de não apenas acumular, mas também de absorver e generalizar o conhecimento (MONARD et al., 1997).

O objetivo do AM, adotado por Zadrozny (2010), é a programação de computadores para aprender automaticamente um determinado comportamento ou padrão através de exemplos (base de dados conhecida) ou observações em que o sistema deve aprender diretamente através de uma interação com o meio ou através de um simulador.

Ainda não há como os computadores extraírem conhecimento tal como os seres humanos, mas foram desenvolvidos algoritmos eficientes em determinadas tarefas que envolvem aprendizado.



**Figura 1:** Esquema da fase de Treinamento no processo de Aprendizado Automático. Fonte: do autor.

Durante o processo de aprendizado do sistema automático, podem-se apontar duas fases: o Treinamento (Figura 1) em que são apresentados exemplos e a partir desses o sistema

consegue utilizá-los para formar uma solução que se aproxime da saída desejada; e Utilização, onde são apresentados exemplos ainda não conhecidos pelos sistemas em que o mesmo deve ter a capacidade de generalizá-lo ou extrair um novo conhecimento (KOERICH, 2008).

Os algoritmos de aprendizado automático podem, então, ser vistos como algoritmos que extraem um padrão de um conjunto de dados e podem ser utilizados como algoritmos de Mineração de Dados (ZADROZNY, 2010), que será detalhado mais a frente.

Mitchell (1997) chama a atenção para o desafio que o aprendizado automático enfrenta para construir sistemas computacionais que automaticamente melhorem com a experiência, sendo assim essencial definir o problema de aprendizado de maneira certa. Ele apresenta uma definição onde cita 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  $E$ .

Aplicando a definição dada por Mitchell (1997) em um caso real, toma-se como exemplo o jogo de damas onde se tem que a tarefa  $T$  é aprender a jogar, o desempenho  $P$  é obtido a partir da percentagem de jogos ganhos contra um adversário e a experiência  $E$  pode ser alcançada quando jogar contra si próprio.

O AM pode ser classificado quanto a diferentes características que possui, como: modo de aprendizado, paradigma de aprendizado, forma de aprendizado, estratégia de aprendizado e linguagem de descrição utilizada para descrever exemplos (MONARD; PATRI, 2005). Neste trabalho serão abordados apenas os paradigmas, formas e estratégias de aprendizado de máquina, pois são os interessantes ao estudo desse trabalho.

### **2.3 Formas de Aprendizado**

O fator mais importante na determinação da natureza do problema de aprendizagem, para Russel e Norvig (2004), é o tipo de realimentação disponível, ou seja, uma formatação ou ajuste nas informações disponíveis que pode ser capaz de fornecer novos dados. Assim, a área de aprendizado de máquina distingue três formas de aprendizado: Supervisionado, Não supervisionado e Semi-supervisionado, também conhecido como Aprendizado por Reforço.

### 2.3.1 Supervisionado

O aprendizado supervisionado utiliza conhecimentos empíricos, habitualmente representados por um conjunto de observações ou exemplos de classes conhecidas, com o qual é possível encontrar uma hipótese que classifique novas situações em algum grupo de classes já estabelecidas (OSÓRIO, 1999).

Seja um táxi guiado por um agente inteligente<sup>1</sup> um exemplo em que se podem ter os seguintes casos:

- 1) Quando o instrutor do agente gritar “Freie”, o agente pode aprender uma regra de condição-ação sobre quando deve frear.
- 2) Ao ver imagens ou elementos que indiquem que há outros veículos perto, o agente poderá reconhecê-los.

Nestes dois casos temos exemplos de aprendizado supervisionado, em que o agente produz uma saída booleana (Verdadeiro ou Falso) de acordo com as informações armazenadas: de frear ou não frear no caso (1) e se o cenário possui ou não um veículo próximo em (2).

### 2.3.2 Não Supervisionado

Nesta forma de aprendizado o objetivo é estabelecer a existência de *clusters* (similaridades em um conjunto de observações) em classes não conhecidas. O comportamento resultante desta forma de aprendizado é usualmente comparado com técnicas de análise de dados empregadas na estatística (OSÓRIO, 1999).

Seguindo ainda o exemplo mencionado anteriormente do agente inteligente, um caso de aprendizagem não-supervisionada seria o desenvolvimento dos conceitos de dias de tráfego bom e dias de tráfego ruim sem ter recebido exemplos identificados deles.

### 2.3.3 Semi-supervisionado ou por reforço

---

<sup>1</sup> Um agente é algo que percebe e age em um ambiente. A função de agente relativa a um agente especifica a ação executada pelo agente em resposta a qualquer seqüência de percepções. (...) Um agente racional age para maximizar o valor esperado da medida de desempenho, dada a seqüência de percepções vista até o momento (Russel e Norvig, 2004, p. 51).

É a forma de aprendizado mais geral das três. Aqui, o usuário possui indicações imprecisas sobre o comportamento final desejado (por exemplo, sucesso ou não sucesso de uma ação). Nesta forma de aprendizado se tem uma avaliação qualitativa do comportamento do sistema, sem mensurar quantitativamente o erro ou desvio do comportamento esperado (OSÓRIO, 1999). O aprendizado por reforço inclui o subproblema de aprender como o ambiente funciona para poder retirar informações necessárias para a criação de alguma regra (RUSSEL; NORVIG, 2004).

Ainda a respeito do agente inteligente do táxi, depois de receber reclamações ou algum indício do descontentamento do passageiro, o agente recebe uma informação de que seu comportamento foi indesejável.

## **2.4 Paradigmas de Aprendizado Automático**

Alguns paradigmas de aprendizado são discutidos constantemente, tais como: o simbólico, estatístico, *instance-based*, conexionista, genético e ainda o baseado em exemplos. Uma breve apresentação do que são esses paradigmas será realizado, mas o foco está no paradigma simbólico, que trata das árvores de decisões usadas neste trabalho.

### **2.4.1 Paradigma Simbólico**

Este paradigma tenta aprender através da construção de representações simbólicas de um conceito por meio da análise de exemplos e contra-exemplos do conceito em questão, onde as representações simbólicas estão na forma de alguma expressão lógica, árvores de decisão, regras de produção ou redes semânticas. As representações mais estudadas são as regras de produção e as árvores de decisão.

Métodos de indução de árvores de decisão a partir de dados empíricos (particionamento recursivo) foram estudados nas áreas de IA e Estatística e surgiram da tradução das árvores de decisão para regras. Mais tarde, surgiram os métodos que induziam regras diretamente a partir dos dados. São exemplos desses métodos os algoritmos ID3 e C4.5 (MONARD et. al, 1997).

### **2.4.2 Paradigma Estatístico**

Os métodos de classificação estatística focam tarefas em que todos os atributos têm valores contínuos ou ordinais, que podem ser também paramétricos, isto é, assumir alguma forma de modelo, encontrando, assim, valores apropriados para os parâmetros do modelo a partir dos dados.

Como exemplo, um classificador linear assume que classes podem ser expressas como combinação linear dos valores dos atributos. O classificador procura, então, uma combinação linear única que forneça a melhor aproximação para o conjunto de dados.

Normalmente os classificadores estatísticos assumem que os valores dos atributos são distribuídos para assim usar os dados fornecidos para determinar médias, variâncias e covariâncias das distribuições (MONARD et. al, 1997).

#### 2.4.3 Paradigma Instance-based

Os métodos *instance-based* (baseado em instância) classificam casos não conhecidos através de casos similares que já são conhecidos, assumindo que o não conhecido assumirá a classe do caso conhecido que se aproxima de suas características.

Estes sistemas devem ter algumas precauções, tais como os casos de treinamento a serem armazenados, pois armazenar todos os casos tornaria o sistema muito lento e de difícil manuseio; como deve ser medida a similaridade entre os casos quando tiver tanto atributos contínuos quanto ordinais e como cuidar de casos que possuem muitos atributos irrelevantes; e como um novo caso deve ser relacionado com os casos já armazenados, onde se pode usar uma simples comparação a um caso armazenado que seja bem próximo do novo caso ou usar vários casos levando-se em consideração diferentes graus de similaridade (MONARD et. al, 1997).

#### 2.4.4 Paradigma Conexionista

Os exemplos mais conhecidos de sistemas conexionistas são as redes neurais, construções matemáticas simplificadas inspiradas no sistema nervoso, nas quais se encontram unidades altamente interconectadas, de onde o nome conexionismo se baseia. Acredita-se que as redes neurais e essas analogias com a biologia possuem potencial para resolver problemas que requerem intenso processamento sensorial humano, tal como visão e reconhecimento de voz.



Muitos autores consideram as redes neurais como métodos estatísticos paramétricos já que o treino de uma rede neural significa encontrar valores apropriados para pesos (parâmetros) (MONARD et. al, 1997).

#### *2.4.5 Paradigma Genético*

Segundo Monard et. al (1997), o paradigma genético é uma representação do modelo biológico que consiste em uma população de elementos classificatórios que competem para chegar a uma solução geral. Assim como na seleção natural de Darwin, os elementos que tem o desempenho mais fraco são eliminados e os que são mais fortes permanecem gerando derivados deles mesmos. Estes sistemas são utilizados para buscar soluções para os problemas de aprendizado de máquina.

Alguns operadores genéticos básicos, citados por Monard et. al, (1997), são utilizados junto aos elementos (indivíduos) da população para gerar novos indivíduos. Esses operadores são:

- Reprodução: este operador produz cópias de um indivíduo, sendo este número proporcional à sua aptidão de sobrevivência.
- Cruzamento: a troca de material genético entre indivíduos.
- Mutação: a mutação é importante para perpetuar uma espécie. Nos algoritmos genéticos, a mutação modifica o valor de uma posição aleatória da cadeia de símbolos que representa um indivíduo.
- Inversão: é a troca de símbolos na cadeia que representa um indivíduo. Os símbolos em posições mais próximas têm maior probabilidade de serem rearranjados dentro da cadeia

#### *2.4.6 Paradigma Baseado em Exemplos*

Aha, Kibler e Albert (1991) citam ainda o Paradigma baseado em exemplos, que são aqueles que podem classificar um exemplo a partir da classe conhecida de outro exemplo já avaliado, atribuindo, assim, a esse novo exemplo a mesma classe do exemplo já conhecido. As principais características desses sistemas são a identificação e retenção de casos prototipados que juntos representem toda a informação necessária para a classificação de

novos casos. Algumas das técnicas mais difundidas são *Nearest Neighbours* (Vizinhos mais próximos) e a do Raciocínio Baseado em Casos (RBC) (SGARBI, 2007).

## **2.5 Estratégias de Aprendizado Automático**

O aprendizado de um novo conhecimento a partir de conhecimentos já disponíveis pode ser realizado de várias maneiras. Assim, o tipo de raciocínio que uma pessoa utiliza para obter novos conhecimentos define uma estratégia de aprendizado que pode também contribuir para um critério de classificação dos processos de aprendizado de máquina.

As estratégias básicas de aprendizado automático podem ser citadas de acordo com a complexidade de inferência apresentadas pelo aprendiz, que por ordem crescente de dificuldade para o aprendizado, temos: aprendizado por hábito, aprendizado por instrução, aprendizado por dedução, aprendizado por analogia e aprendizado por indução (MONARD et. al, 1997).

### **2.5.1 Aprendizado por Hábito**

No aprendizado por hábito não é necessário que o aprendiz realize nenhum raciocínio sobre uma informação dada, pois o conhecimento é assimilado diretamente pelo aprendiz. Dessa forma, o aprendizado é realizado pela memorização direta de descrições de um conceito.

Um exemplo dessa estratégia utilizada em AM é quando um computador possui um algoritmo específico para reconhecer um conceito (MONARD et. al, 1997).

### **2.5.2 Aprendizado por Instrução**

Nesta estratégia, os conceitos são adquiridos de uma fonte, como por exemplo, um livro ou outra pessoa que já possua conhecimento acerca daquele assunto. O aprendiz não copia diretamente a informação fornecida para a memória e sim seleciona os fatos mais importantes e/ou transforma a informação obtida em formas mais apropriadas. (MONARD et. al, 1997).

### **2.5.3 Aprendizado por Dedução**

A dedução é definida como uma inferência logicamente correta, já que a conclusão adquirida parte de outras premissas iniciais tidas como verdadeiras, preservando assim sempre a verdade.

Aqui, o aprendiz obtém um conhecimento que é resultado de uma transformação de um conhecimento já alcançado. No aprendizado por dedução é memorizado o resultado de uma série de deduções ou cálculos sobre a informação reunida que normalmente é realizado (MONARD et. al, 1997).

#### *2.5.4 Aprendizado por Analogia*

Este aprendizado utiliza a definição de analogia para alcançar um novo conceito, quer dizer, buscam-se semelhanças entre conceitos diferentes. Assim, é realizada uma adaptação de uma regra já existente, modificando-a de forma a ser aplicada a um novo conceito.

Monard et al (1997) afirma que tal estratégia pode ser vista como uma combinação da estratégia por indução com a dedutiva, na qual se determinam características gerais unindo os conceitos que estão sendo comparados, através da inferência indutiva e, posteriormente, utilizando-se da inferência dedutiva e das características geradas no processo indutivo para determinar o conceito a ser aprendido.

#### *2.5.6 Aprendizado por Indução*

Os sistemas de aprendizado automático utilizam o princípio da indução (inferência lógica), em que devem analisar informações e absorvê-las a fim de generalizá-las a partir de um conjunto de exemplos. Com isso, um novo conceito é aprendido efetuando-se inferência indutiva sobre os exemplos conhecidos.

A inferência é fundamental para criar novos conhecimentos e prever futuros eventos. O raciocínio da indução é partir de um universo específico para um universo geral. Porém, é importante estar atento às generalizações geradas, pois o conhecimento pode ser de pouco ou nenhum valor.

As hipóteses geradas pela inferência indutiva podem ou não preservar a verdade, levando a conclusões que não atendam aos resultados esperados. Assim, podem ser apontados dois fatores importantes que têm peso durante o processo de indução: o número de informações colhidas antes de se obter uma regra e atributos corretos, relevantes e relacionados entre si,

visto que podem ser escolhidos atributos que não possuem qualquer ligação com outros atributos necessários para se obter uma conclusão (BATISTA, 2003).

Dois dos métodos de aprendizado indutivos muito conhecidos serão apresentados a seguir: o aprendizado AQ e o Dividir para Conquistar.

- **Aprendizado AQ**

Este aprendizado utiliza o método de cobrir progressivamente os dados de treinamento enquanto as regras de decisão são geradas. Seu funcionamento é baseado 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, generalizando, passo a passo as descrições dos exemplos positivos selecionados (MICHALSKY; BRATKO; KUBAT, 1998).

- **Dividir para conquistar (família TDIDT)**

Neste método, o conjunto de exemplos é dividido em subconjuntos, podendo essa estratégia ser aplicada nestes subgrupos e assim recursivamente até que se obtenha um resultado conveniente para se trabalhar com mais facilidade. Dentro do aprendizado Dividir para Conquistar encontramos a família TDIDT (*Top-Down Induction Trees*), da qual fazem parte as Árvores de Decisão (SGARBI, 2007).

### 3 MINERAÇÃO DE DADOS

#### 3.1 Descoberta de Conhecimento em Banco de Dados

Devido ao atual baixo custo para armazenamento de informações e o crescimento da informatização nas empresas, um grande volume de dados é armazenado nos computadores. Apesar da grande quantidade de informação disponível, muitas outras informações estão escondidas e passam despercebidas por seus gerenciadores, conforme afirma Rezende (2005).

Para solucionar esta questão, surgiu o conceito de Descoberta de Conhecimento em Base de Dados (DCBD), originado do termo em inglês *Knowledge Discovery in Databases* (KDD). Rezende (2005) cita ainda que o KDD combine diferentes técnicas oferecidas por diversas áreas, como Banco de Dados, Aprendizado de Máquina, Estatística, Recuperação de Informação, Computação Paralela e Distribuída.

O termo KDD surgiu em 1989 para enfatizar que o conhecimento é o produto final da descoberta orientada a dados. Ele tem por função a procura de padrões em uma massa de dados que pode ser útil ou interessante, revelando um conhecimento até então oculto.

O processo de descoberta de conhecimento abrange diversos fatores, desde o modo como os dados são armazenados e acessados, a escolha do algoritmo para ser usado com grandes quantidades de dados de forma a se obter uma execução eficiente, até como os resultados podem ser interpretados e visualizados e como a interação entre homem-máquina pode utilmente ser modelada e suportada (FAYYAD et al., 1996).

A Descoberta de Conhecimento em Base de Dados pode ser dividida em nove etapas, segundo Fayyad et. al (1996) e em três etapas, segundo Goldschmidt e Passos (2005). O modelo proposto pelo último autor é constituído pelas etapas de Pré-processamento dos dados, a busca pelo conhecimento propriamente dita, chamada de Mineração de Dados (MD), e um Pós-processamento das informações obtidas pela MD.

#### 3.2 Pré-processamento

Esta etapa reúne os passos um a seis, segundo o modelo hierárquico de Fayyad et al. (1996), em que deve ser identificado o objetivo do processo de busca e a compreensão do domínio da aplicação, em que dados ou variáveis devem ser analisados e selecionados para formar o conjunto no qual se realizará o processo de KDD. Após a seleção dos dados ou

variáveis, estes devem ser tratados, eliminando os ruídos ou inconsistências (como campos em branco ou erros cometidos no cadastro).

É feita, então, uma redução e projeção dos dados, aplicando-se recursos úteis para converter as informações para um padrão (por exemplo, criação de categorias para faixa etária e conversão de variáveis nominais em numéricas) e que gere uma melhor agilidade durante o processamento.

E por fim, os dados são ajustados para servirem de entrada para o algoritmo de Mineração de Dados, que deve ser escolhido também nessa etapa. A escolha do algoritmo a ser utilizado deve incluir qual ou quais modelos e parâmetros devem ser apropriados para o objetivo em questão (FAYYAD et al., 1996).

### **3.3 Mineração de Dados**

Por vezes, o termo Mineração de Dados é confundido com o termo KDD. Há autores, como Navega (2002), que consideram os termos sinônimos e define MD como “algoritmos que processam os dados e encontram padrões válidos, novos e valiosos” e ainda “uma hierarquia, algo que começa em instâncias elementares (embora volumosas) e terminam em um ponto relativamente concentrado, mas muito valioso”.

Outros autores como Fayyad et. al (1996, p. 40) consideram a Mineração de Dados como uma etapa da Busca por Conhecimento e a descreve como “passo no processo de KDD que consiste na aplicação de análise de dados e algoritmos de descoberta que produz uma determinada enumeração de padrões (ou modelos) sobre os dados”.

O reconhecimento de padrões<sup>2</sup> implica na simplificação dos dados brutos, considerando aquilo que é genérico e abandonando o que for específico, pois o conhecimento buscado se refere a informações comuns presentes na maioria dos dados (NAVEGA, 2002).

Existem diversos algoritmos que implementam técnicas diferentes para a identificação de padrões em um volume de dados. Algumas dessas técnicas são brevemente descritas abaixo.

#### **3.3.1 Associação**

---

<sup>2</sup> Padrões são unidades de informações que se repetem, ou então são sequências de informações que dispõem de uma estrutura que se repete (NAVEGA, 2002).

A técnica de associação tem o objetivo de descobrir entre os atributos de um conjunto de dados aqueles elementos que ocorrem (ou não) em comum e com determinada frequência no conjunto (GRÉGIO, 2007) e gerar regras (conhecidas como regras de associação) que representem estas associações entre os atributos (BLESSER, 2004).

Para avaliar a credibilidade das regras geradas usam-se duas métricas: suporte e confiança. O suporte é o percentual com que a regra aparece na base de dados e a confiança mede a validade da regras (GRÉGIO, 2007).

O algoritmo busca por regras do tipo ‘Se X ocorre na base de dados, então Y também ocorre’, sendo X antecedente e Y o conseqüente da associação (GRÉGIO, 2007). Um exemplo comum em que podemos verificar regras de associações ocorre em listas de compras. Na regra {pão} → {café}, onde {pão} é o antecedente e {café} o conseqüente e se tem suporte igual a 60 e confiança igual a 30, significa que em 60% dos registros de compras analisados, apareciam os itens *café* e *pão* e que 30% das pessoas que compraram *pão* também compraram *café* (JOST, 2009).

### 3.3.2 Classificação

Consiste em classificar os registros em categorias ou classes pré-definidas por um modelo construído a partir de um grupo de treinamento formado por dados que sirvam de exemplos para classificar os novos registros. Assim, quando há uma inserção o novo registro já é classificado automaticamente (GOLDSCHMIDT; PASSOS, 2005).

Freitas (1998) aponta como diferença entre as técnicas de Associação e a de Classificação, o nível sintático, já que a primeira permite vários atributos em sua conclusão, enquanto a segunda permite apenas um.

A regra de Classificação é utilizada em vários algoritmos, os quais possuem métodos diferentes para a indução do conhecimento. Alguns desses métodos são as redes neurais artificiais (RNA), regras utilizando a condição *IF-THEN* e as árvores de decisão (JOST, 2009), que serão detalhadas mais a frente.

### 3.3.3 Agrupamento

A técnica de agrupamento, também conhecida como Clusterização (*clustering*), é caracterizada por Goldschmidt e Passos (2005) por separar em grupos os registros com características comuns, buscando também uma distinção entre os grupos definidos.

O conceito de classes-modelos utilizadas para a classificação de novos dados não é necessária já que os grupos serão formados após todas as entradas terem sido informadas. O algoritmo deve identificar determinado número de grupos de dados e calcular a associação dos dados de entrada aos grupos de saída.

Os algoritmos de agrupamento necessitam de várias comparações dos dados entre si para se chegar a grupos estáveis. Isso pode gerar altos custos de processamento e tornar a aplicação inviável para uma grande quantidade de dados disponíveis, conclui Grégio (2007).

#### *3.3.4 Regressão*

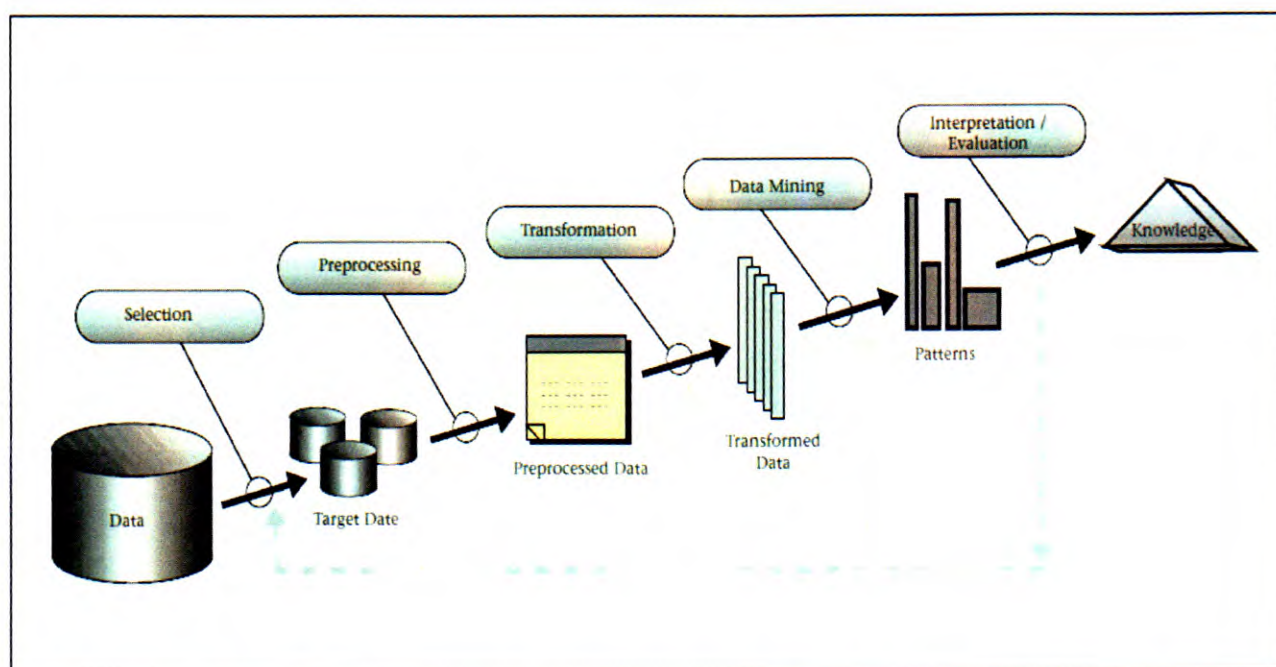
Esta técnica é semelhante à Classificação ao se criar um modelo: ela utiliza exemplos para classificar registros não conhecidos. Segundo Grégio (2007), a diferença está na obtenção dos dados: enquanto a Classificação obtém uma classe discreta para os dados, a Regressão lida com um valor numérico real (valor objetivo) e obtém uma função numérica para o cálculo deste valor como resultado do treinamento do algoritmo.

### **3.4 Pós-Processamento**

Nesta última etapa, que compreende os passos oito e nove de Fayyad et. al (1996), ocorre a interpretação dos padrões encontrados e o uso deste conhecimento, seja diretamente ou incorporando-o a outros sistemas para ação futura ou simplesmente a um relatório para aqueles interessados. A Figura 2 ilustra os passos descritos no processo de KDD.

Apesar de a técnica de MD identificar padrões novos e válidos, eles ainda não são capazes de identificar padrões que sejam valiosos para os clientes. Por isso, o processo de Busca por Conhecimento em Base de Dados ainda requer uma forte interação com analistas, que serão aptos a determinar esses valores nos padrões identificados (NAVEGA, 2002).





**Figura 2:** Passos que compõem o processo de KDD. Fonte: (FAYYAD et al., 1996).

### 3.5 Indução de Árvores de Decisão

A filosofia de funcionamento de algoritmos baseados em Árvores de Decisão (AD) se baseia na estratégia de Dividir para Conquistar, que resumidamente pode ser descrita como uma série de divisões de um problema em problemas menores até encontrar uma solução para cada um desses pequenos problemas (FONSECA, 1994).

Assim, nas árvores de decisão, divide-se o universo do problema sucessivamente em subconjuntos (criando nós contendo os testes para cada situação) até que todos estes estejam classificados em uma classe ou até que uma classe não possua requisitos para ser novamente dividida (neste caso, gerando uma folha contendo a classe mais significativa).

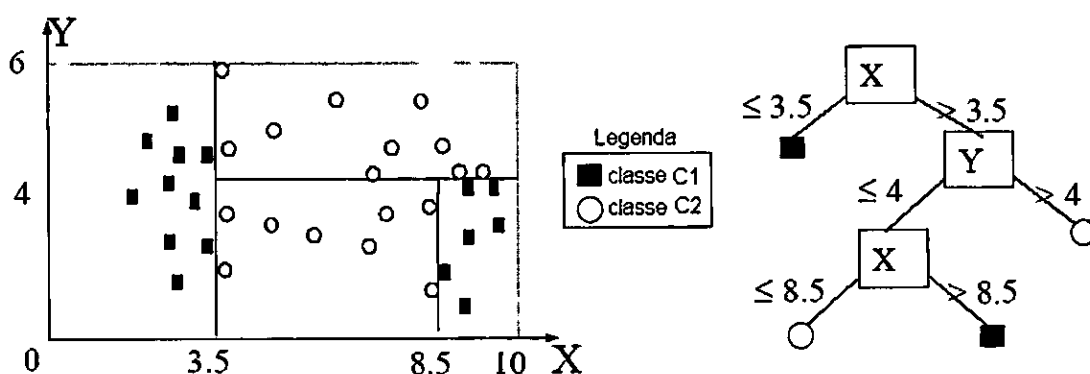
Dentre os aspectos determinantes para a escolha de um algoritmo de construção de uma árvore de decisão há pontos importantes que devem ser destacados: qual o critério de escolha da característica utilizado em cada nó, como calcular as divisões no conjunto de exemplos, qual o melhor momento para decidir que um nó deve ser uma folha e qual critério para selecionar a classe a atribuir a cada folha (FONSECA, 1994).

Algumas vantagens dos algoritmos baseados em árvores de decisão são o fato de eles poderem ser utilizadas com qualquer tipo de dados, além de uma fácil compreensão para o usuário e uma estrutura final do classificador simples (FONSECA, 1994), motivos pelos quais se escolheu trabalhar com os mesmos.

No exemplo ilustrado abaixo (Figura 3), deseja-se encontrar regras que descrevam a situação presente no espaço delimitado pelo retângulo. Para gerar essas regras usou-se uma árvore de decisão.

Segundo Monard e Prati (2005), a estrutura de uma AD é constituída por:

- Um nó folha que corresponde a uma classe (no exemplo da Figura 3, as classes são C1 e C2) ou
- Um nó de decisão que contém um teste para verificar o atributo (no exemplo da Figura 3, os nós de decisão são X e Y) e do qual saem arestas valoradas para uma subárvore. Cada subárvore possui a mesma estrutura da árvore maior.



*Figura 3: Exemplo de uma árvore de decisão. Fonte: (Liu et al., 2000)*

O processo de classificação de um novo exemplo inicia-se pelo nó mais acima da árvore, chamado raiz, onde é realizado o primeiro teste no atributo do novo exemplo. Escolhe-se aquela aresta que for verdadeira para o atributo e os testes seguem em cada nó visitado, passando por aquelas arestas que convêm. O caminho é trilhado até que uma folha seja alcançada e o novo exemplo é atribuído a uma classe (MONARD; PRATI, 2005).

Para demonstrar a construção de uma árvore de decisão, seja o exemplo apresentado por Quilan (1993), onde um conjunto de observações (Tabela 1) especifica quando é um bom dia para se jogar Golf. Neste caso, os atributos considerados são: Tempo, Temperatura, Umidade e Vento; e as classes definidas são Jogar e Não jogar.

A construção de uma AD, descrita por Servente (2002), se dá pelo método de Hunt, a partir de um conjunto  $T$  de treinamento e classes denotadas  $\{C_1, C_2, \dots, C_k\}$ . Há três possibilidades:

1.  $T$  contém um ou mais casos, todos pertencentes à uma única classe  $C_j$ . Neste caso, a árvore conterá apenas uma folha representada pela classe  $C_j$ .

2.  $T$  não possui nenhum caso. A árvore de decisão será de apenas uma folha, porém a classe associada a esta folha deve ser determinada por uma informação que não pertença a  $T$ .

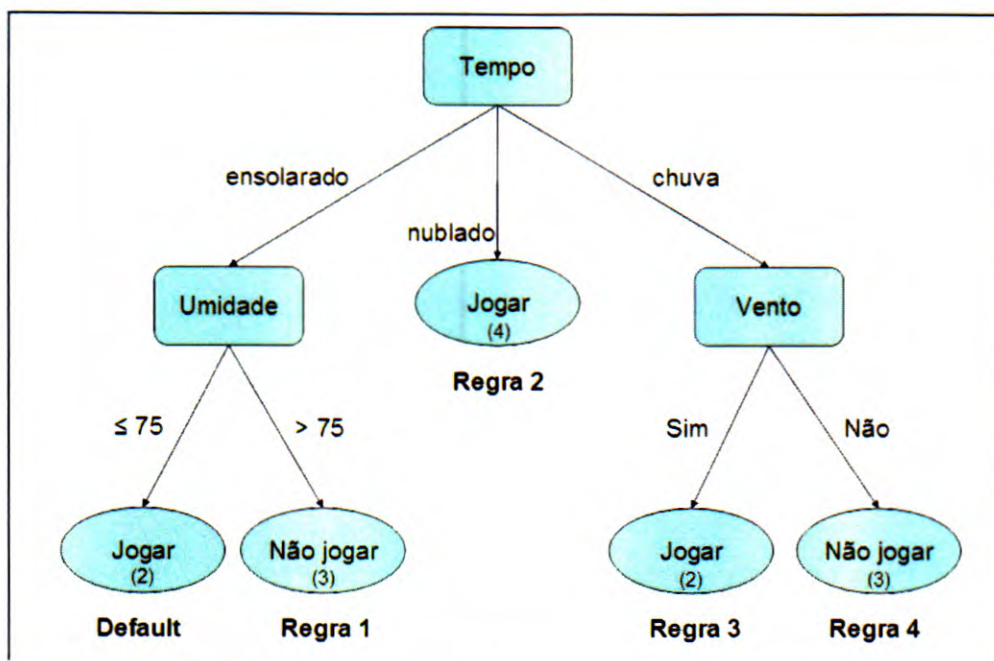
3.  $T$  possui vários casos pertencentes a várias classes. Sendo este o caso, a ideia é dividir  $T$  em subconjuntos que tenham ou pareçam ter uma coleção de casos pertencentes a uma única classe. Um teste é escolhido baseado em um único atributo que tenha um ou mais resultados mutuamente exclusivos denotados por  $\{O_1, O_2, \dots, O_r\}$ .  $T$  é particionado nos subconjuntos  $T_1, T_2, \dots, T_n$ , onde  $T_i$  contém todos os casos de  $T$  que possuam como resultado o valor  $O_i$  para o teste escolhido. A árvore de decisão para  $T$  consiste em um nó identificando o teste e uma aresta para cada um dos resultados possíveis. Os passos descritos se aplicam recursivamente a cada subconjunto de exemplos de treinamento de maneira que as arestas de cada nó levam para as subárvores construídas a partir do subconjunto de exemplos  $T_i$ .

Tempo	Temperatura	Umidade	Vento	Jogar / Não jogar
Ensolarado	85	85	Não	Não jogar
Ensolarado	80	90	Sim	Não jogar
Nublado	83	78	Não	Jogar
Chuva	70	96	Não	Jogar
Chuva	68	80	Não	Jogar
Chuva	65	70	Sim	Não jogar
Nublado	64	65	Sim	Jogar
Ensolarado	72	95	Não	Não jogar
Ensolarado	69	70	Não	Jogar
Chuva	75	80	Não	Jogar
Ensolarado	75	70	Sim	Jogar
Nublado	72	90	Sim	Jogar
Nublado	81	75	Não	Jogar
Chuva	71	80	Sim	Não jogar

*Tabela 1: Condições para se jogar ou não Golf. Fonte: do autor.*

A árvore de decisão criada para representar o conjunto de observações (Tabela 1) é ilustrada a seguir (Figura 4):





**Figura 4:** Árvore formada para o conjunto de observações da Tabela 1. Fonte: do autor.

Na construção da árvore, qualquer atributo pode ser escolhido para ser raiz da árvore, podendo ser geradas, assim, diferentes árvores para o mesmo conjunto de exemplos. Porém, algumas dessas árvores serão mais complexas e ramificadas que as outras e conseqüentemente regras mais complexas também. Nos problemas reais, em que se tem um grande número de atributos e observações, é importante gerar o mínimo de regras possíveis e que sejam concisas (MONARD et al., 1997).

Para avaliar qual o melhor atributo (o mais relevante) a ser utilizado como raiz da árvore é necessário se utilizar um método. Existem diversos métodos para isso, como razão de ganho, que seleciona o atributo ponderando o ganho de informação esperado em relação ao nó pai (MONARD; PRATI, 2005) e a entropia, que mede a desordem das informações e retorna o atributo de maior organização (e menor entropia) (MONARD et al., 1997).

### 3.6 O algoritmo C4.5

Algoritmo pertencente à família TDIDT (mencionada no capítulo anterior), o C4.5 (Figura 5) constitui uma ferramenta de MD de mesmo nome. Sua finalidade é a mineração de dados utilizando a técnica de classificação para gerar árvores de decisão e regras de classificação (Figura 6) a partir de uma base de dados qualquer.

Por ter sido disponibilizado por seu autor em Quilan (1993), este algoritmo teve grande aceitação do meio acadêmico e é reconhecido não só por sua rapidez de

processamento, como também por sua portabilidade, podendo ser rodado em diferentes sistemas operacionais (BLESSER, 2004).

O C4.5 é uma extensão (melhora) do algoritmo ID3, que se diferencia deste último principalmente por permitir trabalhar com valores contínuos (diversos valores podem ser assumidos) para atributos, além dos valores discretos (quantidade finita de valores que podem ser assumidos), com os quais o ID3 já trabalhava. Os valores discretos dos atributos são separados em dois ramos (ou arestas): um para aqueles em que  $A_i \leq N$  e outra para aqueles que  $A_i > N$ .

```

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 5: Pseudocódigo do algoritmo C4.5. Fonte: (SGARBI, 2007).*

Para realizar a divisão dos dados na árvore de decisão, o algoritmo considera todas as provas possíveis para dividir estes dados e seleciona aquela prova que resulta no maior ganho de informação (método para avaliar o melhor atributo). Para cada atributo discreto, é considerada uma prova com  $n$  resultados, sendo  $n$  o número de valores possíveis que o atributo pode adquirir. E para cada atributo contínuo se realiza uma prova binária sobre cada um dos valores que o atributo adquire nos dados (SERVENTE, 2002).

O algoritmo C4.5 também permite trabalhar com valores desconhecidos para alguns atributos (JOST, 2009). De acordo com Sgarbi, (2007, p. 34), “um caso com um valor desconhecido se divide em fragmentos cujos pesos são proporcionais às frequências relativas, dando por resultado que um caso pode seguir múltiplos caminhos na árvore.”

```

user@calixto[2] /usr/local/R8/Src/c4.5rules -f golf
C4.5 [release 8] rule generator Thu Dec 27 14:24:23 2001
-----
Read 14 cases (4 attributes) from golf
-----
Processing tree 0
Final rules from tree 0:

Rule 2:
  Condicoes do Dia = <coberto
  -> class Jogar [70.7%]

Rule 4:
  Condicoes do Dia = <chuvoso
  Vento = falso
  -> class Jogar [63.0%]

Rule 1:
  Condicoes do Dia = <ensolarado
  Umidade > 75
  -> class Nao Jogar [63.0%]

Rule 3:
  Condicoes do Dia = <chuvoso
  Vento = verdadeiro
  -> class Nao Jogar [50.0%]

Default class: Jogar

Evaluation on training data. (14 items):

Rule      Size  Error  Used      Wrong      Advantage
-----
  2         1    29.3%   4         0 (0.0%)    0 (0|0)    Jogar
  4         2    37.0%   3         0 (0.0%)    0 (0|0)    Jogar
  1         2    37.0%   3         0 (0.0%)    3 (3|0)    Nao Jogar
  3         2    50.0%   2         0 (0.0%)    2 (2|0)    Nao Jogar

Tested 14, errors 0 (0.0%)  <<

      (a) (b)      <-classified as
      ----
          9          (a): class Jogar
           5          (b): class Nao Jogar

```

*Figura 6: Regras geradas pelo C4.5 para o problema Jogar Golf. Fonte: (BERNARDES, 2001).*

Na construção da árvore de decisão, a divisão dos dados pelo método recursivo dividirá o conjunto de treinamento até que um subconjunto tenha somente uma classe ou até que a prova não ofereça melhora. Devido a este fato, a árvore resultante para descrever o conjunto de exemplos normalmente se torna mais complexa que o necessário (QUILAN, 1995).

Para que isto não ocorra, o C4.5 trabalha com poda. Segundo Jost (2009), a poda consiste em substituir uma subárvore com grande taxa de erro por um nó ou aresta, antes ou

depois da árvore ser elaborada. Assim, a árvore é simplificada e facilita a inserção de novos nodos e subárvores, conforme o crescimento do conjunto de dados.

### 3.7 See5/C5.0

Recentemente, foi criado um novo algoritmo, o See5/C5.0 (See5 para Windows, C5.0 para Unix/Linux). O See5/C5.0 é uma extensão do C4.5 com uma série de melhorias, cuja algumas são (QUILAN, 2009):

- Velocidade – o novo algoritmo é significativamente mais rápido que o C4.5
- Uso de memória – o See5/C5.0 consome mesma memória que o C4.5
- Menores árvores de decisão – os resultados do See5/C5.0 são similares aos do C4.5, porém com árvores de decisão menores
- Precisão – os conjuntos de regras gerados pelo See5/C5.0 têm taxas de erros menores
- Reforço – promove melhoras nas árvores e lhes dá mais precisão
- Ponderação – o See5/C5.0 permite atributos de diferentes pesos
- Mineração – uma opção automática do See5/C5.0 minera os atributos para remover aqueles que podem ser inúteis
- Novos tipos de dados – vários novos tipos de dados além dos disponíveis no C4.5 e facilidades para definição de novos atributos em função de outros atributos
- Facilidade de uso – além de algumas opções que foram simplificadas e ampliadas, os programas para gerar árvores (C4.5) e gerar regras (C4.5rules) foram unidos em um único programa

O sistema ABC+ utilizado neste trabalho é voltado para o algoritmo C4.5. Por este motivo, utilizou-se o mesmo no simulador desenvolvido, embora a descoberta da versão See5/C5.0 venha a proporcionar um aumento significativo na geração de resultados do sistema ABC+. Essa proposta fica como dica para projetos futuros.

## 4 DOMÓTICA

A palavra Domótica originou-se do latim *Domus* que significa casa e Robótica que se refere ao controle automatizado de algo (ALVES; MOTA, 2003).

Bolzani (2010, p.31) define domótica como:

É a ciência moderna de engenharia das instalações em sistemas prediais. A domótica é uma ciência multidisciplinar que estuda a relação entre o homem e a casa. A imersão de pessoas em ambientes computacionalmente ativos revelou a necessidade do uso de técnicas mais sutis que gerenciassem a complexidade e o dinamismo das interações dos moradores com o ambiente residencial saturado de diminutos dispositivos eletrônicos interligados em rede.

Assim, podemos concluir que a domótica surgiu para aprimorar o ambiente residencial tornando-o mais adaptável (adequado) aos seus habitantes, facilitando suas atividades rotineiras e ao mesmo tempo oferecendo segurança para uma qualidade de vida maior.

A transformação de casas comuns em casas em que se detém o controle de sua iluminação, aquecimento e ventilação, sistemas de irrigação, temperatura e controle da água, dentre outros sistemas que se podem controlar, é hoje realidade em muitas habitações de vários países (ALVES; MOTA, 2003).

Porém, o termo Domótica tem perdido popularidade para dar lugar ao termo Casa ou Residência Inteligente, por este ter mais apelo de *marketing* e sugerir um conceito mais amplo, que vai além do contexto de uma residência e abrange praticamente qualquer ocasião de interatividade do homem com o ambiente (BOLZANI, 2010).

Uma residência automatizada possui um sistema domótico que integra todos os dispositivos para automatizar e controlar a residência. O sistema domótico é composto por vários elementos, entre eles os atuadores<sup>3</sup>, sensores<sup>4</sup> e controladores<sup>5</sup>, que compõem a parte física do sistema; a rede de dados ou rede domótica que constitui o barramento (ou

---

<sup>3</sup> Atuadores são dispositivos eletromecânicos que têm suas características alteradas conforme os impulsos elétricos recebidos. São exemplos de atuadores lâmpadas, sirenes, fechaduras eletromagnéticas, portões eletrônicos e eletrodomésticos em geral. (BRETERNITZ, 2001)

<sup>4</sup> Sensores são dispositivos que transformam parâmetros físicos, como temperatura, umidade, entre outros, em sinais elétricos apropriados para que os sistemas domóticos possam analisá-los. Exemplos de sensores são: sensores de temperatura, umidade, de intensidade de iluminação, detector de movimento e fumaça, etc. (BRETERNITZ, 2001)

<sup>5</sup> O controlador é o elemento central que gerencia o sistema domótico. Normalmente os outros elementos, como os sensores e os atuadores, se conectam ao controlador enviando e recebendo informações. Nele podem estar as interfaces de usuário, as quais são necessárias para apresentar e receber informações (via teclado, monitor e outros). (BRETERNITZ, 2001)



cabeamento), o qual permite realizar a comunicação entre os diferentes dispositivos existentes no sistema domótico; e a interface com o usuário (BRETERNITZ, 2001).

#### 4.1 História da domótica

Pode se tomar como ponto inicial da ideia de automatizar residências quando nos anos 70, os sistemas AVAC (aquecimento, ventilação e ar-condicionado) passaram a ser controlados eletronicamente nos edifícios. Tal sistema era controlado através de pequenos chips ligados a sensores instalados no ambiente onde atuavam de maneira rápida, realizando alterações na condição climática desses edifícios (ALVES; MOTA, 2003).

Nos anos 80, os sistemas de segurança, iluminação e intrusão também passaram a ser automatizados, havendo agora uma organização entre os componentes desse sistema. *Lloyds Building*, Londres, projetado pela *Richard Roger Partnership*, foi o primeiro edifício construído a utilizar o conceito de Edifícios Inteligentes. Apesar de possuir avançados sistemas tecnológicos, faltava-lhe uma coisa: integração entre eles.

Outros edifícios que foram construídos posteriormente também nos anos 80, como o *NEC Tower* construído pela *Nikken Sekkei*, o *IBM Century Tower* construído pela *Foster Associates Ltd*, e mais recentemente o *Turbine Tower Building* construído pela *Richard Rogers*, adotaram a mesma ideia do *Lloyds Building*. E hoje a Domótica é uma realidade em diversos países (ALVES; MOTA, 2003).

#### 4.2 Domótica Inteligente

Os termos automação residencial, casas inteligentes, domótica inteligente, entre outros, têm sido banalizados ou utilizados incorretamente como sinônimos. A automação residencial é um ramo da automação predial especializada no controle de operações em um espaço doméstico. Ela se utiliza de sistemas de controle para gerenciar equipamentos (eletroeletrônicos e eletromecânicos), reduzindo a necessidade de interação humana através do uso de sensores que coletam informações e analisam seus parâmetros, possibilitando a tomada de uma decisão que pode disparar uma ação que altere o estado de um atuador no ambiente.

A residência inteligente não deve ser considerada inteligente apenas por automatizar o funcionamento de seus dispositivos através da automação residencial. O processo de

automação e controle é um meio para a criação de serviços e aplicações que em conjunto tornarão uma casa próxima ao conceito de inteligente (BOLZANI, 2010).

Não existe ainda uma definição concisa para Casa Inteligente, mas há um consenso de funções que uma residência inteligente deve desempenhar: ajudar seus habitantes a viverem de forma saudável, feliz e segura; executar várias tarefas automaticamente para diminuir o trabalho rotineiro em uma casa; deve prover um maior nível de autonomia pessoal e vida independente; deve utilizar eficazmente e minimizar o uso de recursos naturais; e seguir o conceito de computação pervasiva<sup>6</sup> e ubíqua<sup>7</sup>. Mas a principal característica que se pode destacar em uma casa inteligente é integrar atividades de trabalho, aprendizado e diversão (BOLZANI, 2010).

Segundo Tonidadel, Takiuchi e Melo (2004), o termo automação residencial tem que evoluir para o conceito de Domótica Inteligente, que deve ir além de um sistema centralizado e dedicado e seu foco deve ir além dos sensores e atuadores. Ela deve ser uma arquitetura descentralizada e adaptativa às necessidades dos habitantes e às resoluções de problemas e gerenciamento de recursos.

Deve-se entender, então, por Domótica Inteligente o processo de automação que incorpore algum mecanismo automático de tomada de decisão baseada em técnicas de Inteligência Artificial. Essa é a principal diferença entre domótica e domótica inteligente: o foco e o paradigma de cada um, onde na domótica inteligente o sistema é que se preocupa em mudar sua atuação de acordo com as necessidades e comportamento de seus habitantes (TONIDADEL; TAKIUCHI; MELO, 2004).

### 4.3 Sistema ABC+

O sistema ABC+ (Automação Baseada em Comportamento) (SGARBI; TONIDADEL, 2006a) é uma melhora do sistema ABC proposto por Tonidadel, Takiuchi e Melo (2004). Este sistema, o ABC, atua conforme o comportamento observado dos habitantes de uma residência, o contrário do que acontece em outros sistemas domóticos em que o habitante é quem cria e insere regras sobre suas preferências no sistema.

---

<sup>6</sup> Computação pervasiva tem por objetivo prover a integração do computador no ambiente de forma invisível para o usuário. O computador tem a capacidade de obter informações do ambiente no qual ele está embarcado e controlar, configurar e ajustar a aplicação para melhor atender as necessidades de um dispositivo ou do usuário. (ARAUJO, 2003)

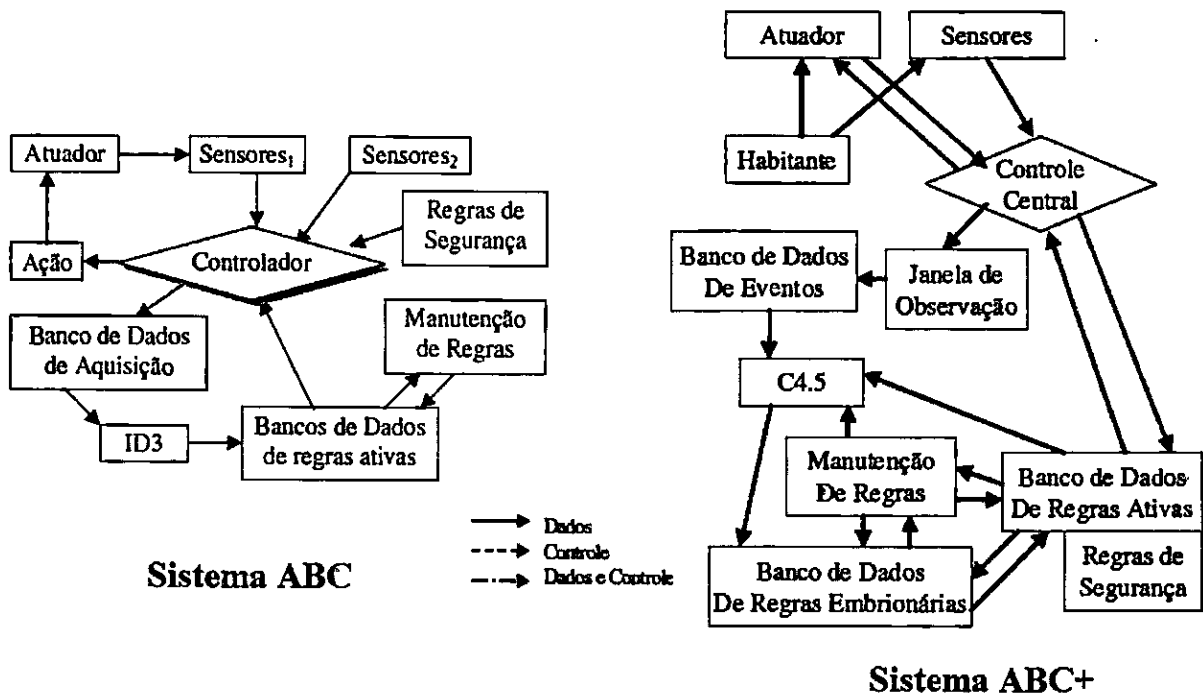
<sup>7</sup> Computação ubíqua se beneficia dos avanços da computação móvel e da computação pervasiva. Ela surge da necessidade de se integrar mobilidade com a funcionalidade da computação pervasiva. (ARAUJO, 2003)

Ambos os sistemas trabalham com a existência de sensores (como os de presença, luminosidade, medidores de temperatura e outros), atuadores (interruptores de luz, ar-condicionado, entre outros), bancos de dados e outros elementos necessários para a criação e controle de regras.

Entretanto, Sgarbi (2007) ressalta algumas deficiências que o ABC apresenta, como: não detectar sequências casuais de eventos no tempo. Por exemplo, se um evento de atuador é disparado depois de pouco tempo ou muito tempo que um evento de sensor ocorre, isto não é avaliado, podendo ser criada, assim, uma regra falsa, pois neste caso não se leva em conta o tempo transcorrido entre um evento de atuador e um evento de sensor.

Outra deficiência identificada é que o primeiro sistema fundamentava-se no algoritmo ID3 (versão anterior do algoritmo C4.5) para a criação das regras de adaptação do sistema de automação. Com o ID3, porém, não é possível trabalhar com variáveis com valores contínuos, apenas com variáveis com valores discretos, o que limitava o trabalho com os valores retornados por alguns sensores, como, por exemplo, o de medida de temperatura e medida de umidade. Além disso, as regras criadas pelo ID3 eram diretamente inseridas no banco de dados de regras, sem verificar se aquele evento poderia ser agradável ou não para o habitante.

Para corrigir as limitações do ABC, o novo sistema ABC+ foi proposto com algumas mudanças em relação ao seu antecessor. Na figura 7, que representa as arquiteturas desses dois sistemas, podem-se perceber algumas diferenças.



*Figura 7: Arquiteturas dos sistemas ABC e ABC+. Fonte: (SGARBI; TONIDADEL, 2006a).*

As principais diferenças entre os dois sistemas são a utilização de uma janela de observação de eventos, um novo conjunto de regras denominadas Regras Embrionárias, o novo processo de desenvolvimento e manutenção das regras e a substituição do ID3 pelo algoritmo C4.5.

As razões para se ter escolhido trabalhar com o C4.5 foi que, além de ser um algoritmo bastante conhecido e adotado para induzir regras a partir de árvores de decisão, ele permite que se trabalhe com valores contínuos para os atributos e com valores de atributos desconhecidos também (SGARBI, 2007).

No sistema ABC+, para cada atuador deve existir uma estrutura paralela de bancos de dados, onde cada atuador é associado a três bancos de dados. Esses bancos de dados são: banco de Eventos, bancos das Regras Ativas, que também guardam as regras de segurança da residência (estas regras são modificadas apenas quando manipuladas diretamente no sistema) e o banco de Regras Embrionárias, que serão detalhados mais a frente.

Assim, a arquitetura do ABC+ é composta por três bancos de dados, a janela de observações de eventos, o C4.5, a lógica de manutenção de regras, que atua para estabelecer quais regras ficam em quais bancos, e o controle central, o qual possui a lógica central do sistema (SGARBI, 2007).

A lógica de funcionamento do sistema é a seguinte: quando algum sensor ou atuador sofre uma mudança de estado pela ação do habitante ou pela natureza, um evento é gerado. Se esse evento for do atuador, a lógica da janela de observação irá analisar o evento e ele poderá ser armazenado no banco de Eventos ou não. Se em determinada ocasião houver um número pré-determinador de eventos armazenados, então o banco de dados Eventos é inserido no algoritmo para indução de regras, o C4.5, que irá criar regras a partir dos dados presentes no banco Eventos. As regras são, então, armazenadas no banco de dados de Regras Embrionárias, onde posteriormente podem ser validadas e transferidas para o banco de dados das Regras Ativas.

Entretanto, se o evento for pertencente a algum sensor, é feita uma busca no banco de dados das Regras Ativas para verificar se há alguma regra relacionada à mudança daquele sensor. Se uma regra condizente à situação do(s) sensor(es) for encontrada, uma ação é então executada (SGARBI, 2007).

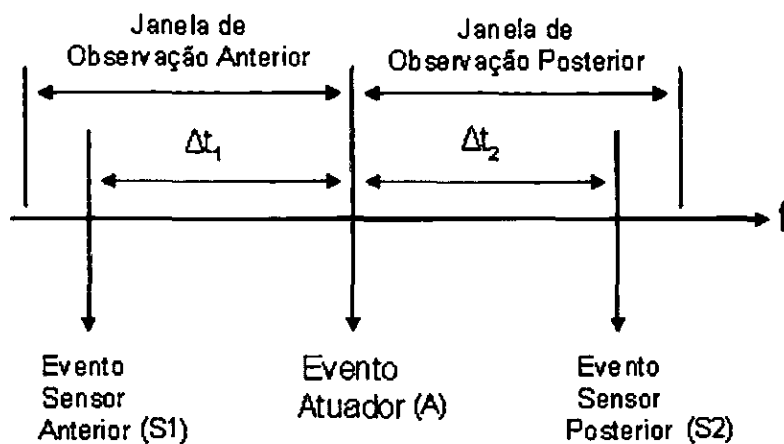
#### *4.3.1 Janela de Observação*

A ideia de criar uma janela de observação surgiu para corrigir uma das deficiências encontradas no sistema ABC: não detectar sequências casuais de eventos no tempo.

Para melhor detalhar o funcionamento do sistema ABC+, se tomará como exemplo um quarto de uma casa onde se tem um sensor para captar a entrada de um habitante no cômodo, outro sensor para detectar a saída do cômodo e um atuador para ligar ou desligar uma lâmpada (SGARBI, 2007).

Quando um habitante entra no quarto e no mesmo instante acende a lâmpada ou quando sai e imediatamente desliga a lâmpada, uma regra notável seria: SE habitante entra ENTÃO acende lâmpada ou SE habitante sai ENTÃO desliga lâmpada. Porém, se o habitante entrar no quarto e após meia hora ou uma hora ele acende a lâmpada, apesar dos sensores detectarem as mesmas informações da primeira situação, as regras acima não devem ser aplicadas, pois há diferenças do ocorrido.

Por isso, a janela de observação se torna fundamental para a criação de regras. Ela consiste em armazenar e comparar cada evento anterior e posterior ao evento a ser analisado, incluindo os horários de cada ação, como ilustra a figura 8 (SGARBI, 2007).



**Figura 8:** Janela de observação do Sistema ABC+. Fonte: (SGARBI, 2007).

Os eventos a serem analisados são eventos de atuadores em que estes mudam de estado, uma lâmpada, como no exemplo dado. Os eventos de sensores, entretanto, não são utilizados para criar regras, mas sim para ativarem uma regra, se esta existir. Esses eventos de sensores podem ser os eventos anterior e posterior na janela de observação, já que servirão apenas para observação e não para gerar regras.

Na janela de observação, cada evento de atuador e seus respectivos eventos de sensor anterior e posterior são tratados isoladamente (um evento de atuador não tem vínculo com o próximo evento de atuador). O objetivo é identificar uma causa (evento no sensor) e um efeito (evento no atuador). Se não houver uma causa e um efeito próximos no tempo, não há sentido para armazenar tal evento para se gerar uma futura regra. Neste caso, uma causa e um efeito com uma notável diferença de tempo seria uma casualidade e casualidades devem ser descartadas para gerar uma regra. O que se busca são eventos repetitivos (padrões ou rotinas) que demonstrem as preferências dos habitantes (SGARBI, 2007).

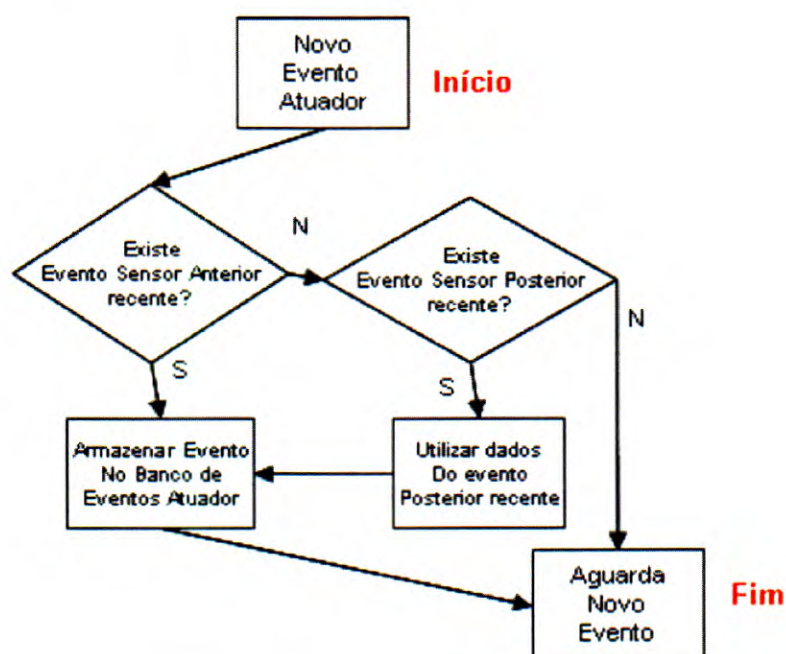
Para melhor entender a lógica da janela de observação (Figura 9), seja o seguinte exemplo: um habitante entra no quarto e dispara um evento de sensor (S1), que é armazenado temporariamente com seu devido horário. Após algum tempo ele acende a lâmpada, gerando um evento de atuador (A). O evento (A) é então armazenado e comparado ao evento anterior (S1). Se (S1) aconteceu dentro de um determinado intervalo de tempo ( $\Delta t_1$ ) menor do que o valor da janela de observação, então significa que o evento atuador (A) está vinculado ao evento anterior ( $\Delta t_1 < \text{janela de observação Anterior}$ ). Neste caso, o evento de atuador é armazenado.

Quando existe um vínculo entre um evento de atuador e um evento de sensor anterior não se realiza uma verificação com o evento posterior (S2) e o sistema volta à situação de

início. Contudo, se não houver esse vínculo com um evento anterior, é necessária a análise com o evento posterior. Assim, o horário do evento atuador é guardado e espera-se a ocorrência de um novo evento (SGARBI, 2007).

Ocorrendo um evento posterior (S2), compara-se a diferença de horários entre (S2) e o evento atuador ( $\Delta t_2$ ). O mesmo processo de verificação realizado com o evento anterior é repetido para o evento posterior: se o evento posterior aconteceu dentro de determinado intervalo de tempo referente à janela de observação, então significa que o evento atuador está vinculado ao evento posterior ( $\Delta t_2 < \text{janela de observação Posterior}$ ) e deve ser armazenado para gerar uma futura regra. Caso contrário, não existe vínculo e ele é descartado.

O limite da janela de observação são os períodos de tempo anterior e posterior. Se o tempo da causa e do efeito não estiverem dentro deste limite, o evento é considerado uma casualidade e não é armazenado (SGARBI, 2007).



**Figura 9:** Fluxograma da lógica da Janela de Observação. Fonte: (SGARBI, 2007).

#### 4.3.2 Regras Embrionárias

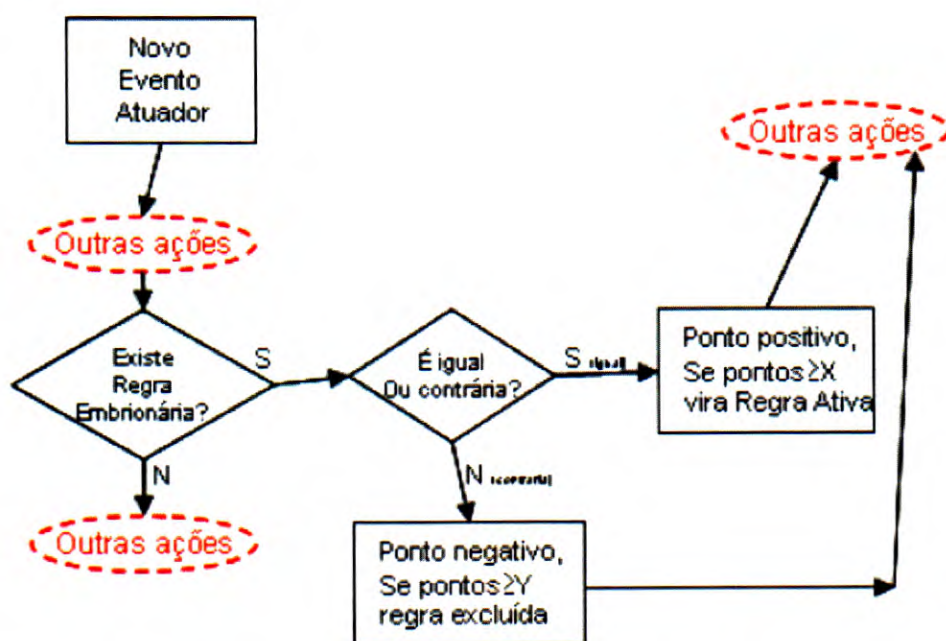
No sistema ABC+ deve existir uma etapa de validação de regras antes que elas sejam classificadas como regras ativas. Toda vez que novas regras são criadas, elas são guardadas inicialmente no banco de Regras Embrionárias. Essa etapa de validação foi criada para solucionar outro defeito encontrado no sistema ABC: a inserção imediata das regras obtidas



através do ID3 no banco de regras ativas, sem nenhuma verificação da validade daquelas regras.

Com a validação proposta para o sistema ABC+, sempre que o habitante gerar condições dos sensores iguais às condições de uma Regra Embrionária, aquela regra vai ganhando valores positivos. Quando o valor de uma regra embrionária atinge determinada pontuação, ela passa para o banco de Regras Ativas.

Se o contrário acontecer, se o habitante contrariar uma regra, ela ganha pontos negativos e ao atingir certa pontuação, a regra é eliminada do banco de Regras Embrionárias (Figura 10) (SGARBI, 2007).



**Figura 10:** Fluxograma da lógica de validação da Regra Embrionária. Fonte: (SGARBI, 2007)

“Outras ações” se referem a um bloco macro com lógicas que realizam outras ações presentes no sistema proposto, mas que não se encontram detalhadas aqui (SGARBI, 2007).

#### 4.3.3 Desenvolvimento e Manutenção das Regras

Segundo Sgarbi e Tonidadel (2006b, p. 3),

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+, identifica-se dois tipos de regras: as Ativas e as Embrionárias. Para o desenvolvimento e manutenção destas regras, são utilizados três bancos de dados para cada atuador existente na residência: o banco de Eventos (tab. 2), o banco de Regras Embrionárias (tab. 3) e o banco de Regras Ativas (tab. 4).

Nas tabelas abaixo são detalhados os campos existentes em cada um dos bancos utilizados pelo sistema ABC+.

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 2: Exemplo de um banco de Eventos. Fonte: (SGARBI, 2007).*

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 3: Exemplo de um banco de Regras Ativas. Fonte: (SGARBI, 2007).*

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
....	....	....	....	....	....	....	....

*Tabela 4: Exemplo de um banco de Regras Embrionárias. Fonte: (SGARBI, 2007).*

No decorrer do texto serão utilizadas as definições a seguir:

- 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.

O processo de classificação das regras (Figura 11) pode ser facilmente entendido. Quando um evento de atuador é vinculado a um evento de sensor durante a etapa de validação na Janela de Observação, este novo evento é armazenado no banco de dados de Eventos. Ao atingir um número pré-determinado de eventos, todas as tuplas presentes no banco de Evento são inseridas no algoritmo C4.5 para que este gere regras a partir do conjunto de observações fornecidos (SGARBI, 2007).

As regras geradas pelo C4.5 são comparadas às regras presentes nos outros bancos, de Regras Embrionárias e Ativas, para que as regras repetidas possam ser eliminadas. Aquelas regras que não forem repetidas vão para o BDEmbrio, onde existem os campos OK e NOK que servem para controlar a validação de cada regra. Sempre que uma regra passa a fazer parte do BDEmbrio, seus campos OK e NOK têm valor zero.

Toda vez que um evento de atuador ocorre, verifica-se primeiramente se existe alguma regras sendo contrariada no BDAtivas. Não havendo regra sendo contrariada, o evento é comparado com as Regras Embrionárias. Se existir alguma regra neste banco relacionada ao evento em questão, o campo OK desta regra é incrementado em um. Caso ocorra o contrário e uma regra no BDEmbrio seja contrariada, o campo NOK é incrementado em um. Quando o campo NOK atinge certo valor, a Regra Embrionária é removida do BDEmbrio.

Os campos OK e NOK funcionam como porcentagens de erros e acertos. Se o valor do campo OK para que uma regra seja validada é oito, por exemplo, e o valor do campo NOK para a regra ser excluída é dois, significa que quando se tem 80% ou mais de acertos, a regra é validada e quando se tem 20% ou mais de erros, a regra é excluída (SGARBI, 2007).

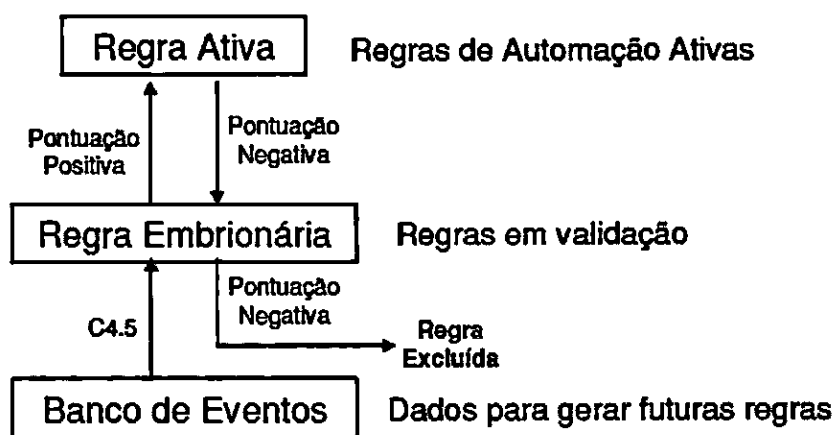
O BDAtivas também possui uma lógica de manutenção. Além das regras a serem executadas na casa, nesse banco também ficam armazenadas as Regras de Segurança, como

'SE existir fogo, ENTÃO cortar energia e gás'. As Regras de Segurança não sofrem modificações justamente por se tratar da segurança do ambiente.

Toda vez que um evento de sensor ocorrer, é verificado se existe alguma regra no BDAtivas condizente com aquela situação. Se for encontrada alguma regra, então ela deve ser executada e, assim, uma ação é realizada no atuador para o qual aquela regra existe e soma-se um ao campo ATIV do BDAtivas (SGARBI, 2007).

Existe uma quantidade limitada de regras que podem fazer parte do banco de Regras Ativas e elas estão ordenadas de acordo com o valor do campo ATIV. Quando se deseja inserir uma regra neste banco e ele está no seu limite, então uma regra terá que ser removida. Aquela regra que tiver o menor valor no campo ATIV é escolhida para ser retirada do banco e voltar a fazer parte do banco de Regras Embrionárias.

Por outro lado, se um evento de atuador contrariar alguma regra do BDAtivas, então é somado um ao valor do campo EXC da regra que foi contrariada. Quando o valor do campo EXC ultrapassar um valor pré-definido, esta regra é excluída do BDAtivas (SGARBI, 2007).



**Figura 11:** Esquema da lógica de Manutenção das Regras. Fonte: (SGARBI, 2007)

O processo de manutenção das regras também se preocupa quando regras do BDEmbrio e BDAtivas ficam velhas. Para isso, quando novas regras são criadas pelo C4.5, subtrai-se um do valor do campo ATIV e do valor do campo OK de todas as regras existentes no BDAtivas e no BDEmbrio. As Regras Ativas que tiverem seu campo ATIV abaixo de certo valor serão rebaixadas ao BDEmbrio e as Regras Embrionárias que tiverem seu campo OK abaixo de certo valor serão excluídas.

Os campos OK, NOK, ATIV, EXC e a quantidade de eventos que devem formar o conjunto de observação a serem inseridos no algoritmo C4.5 funcionam como controladores

para que regras sejam criadas, promovidas ou eliminadas. Estes valores pré-determinados podem, na verdade, ser configurados através de variáveis (SGARBI, 2007).

O esquema de funcionamento do Sistema ABC+ (Figura 12) reflete as rotinas descritas. No anexo A podem ser encontrados algoritmos das ações realizadas quando ocorre um evento de sensor e um evento de atuador e ainda a representação do funcionamento do ABC+ em Rede de Petri (anexo B).

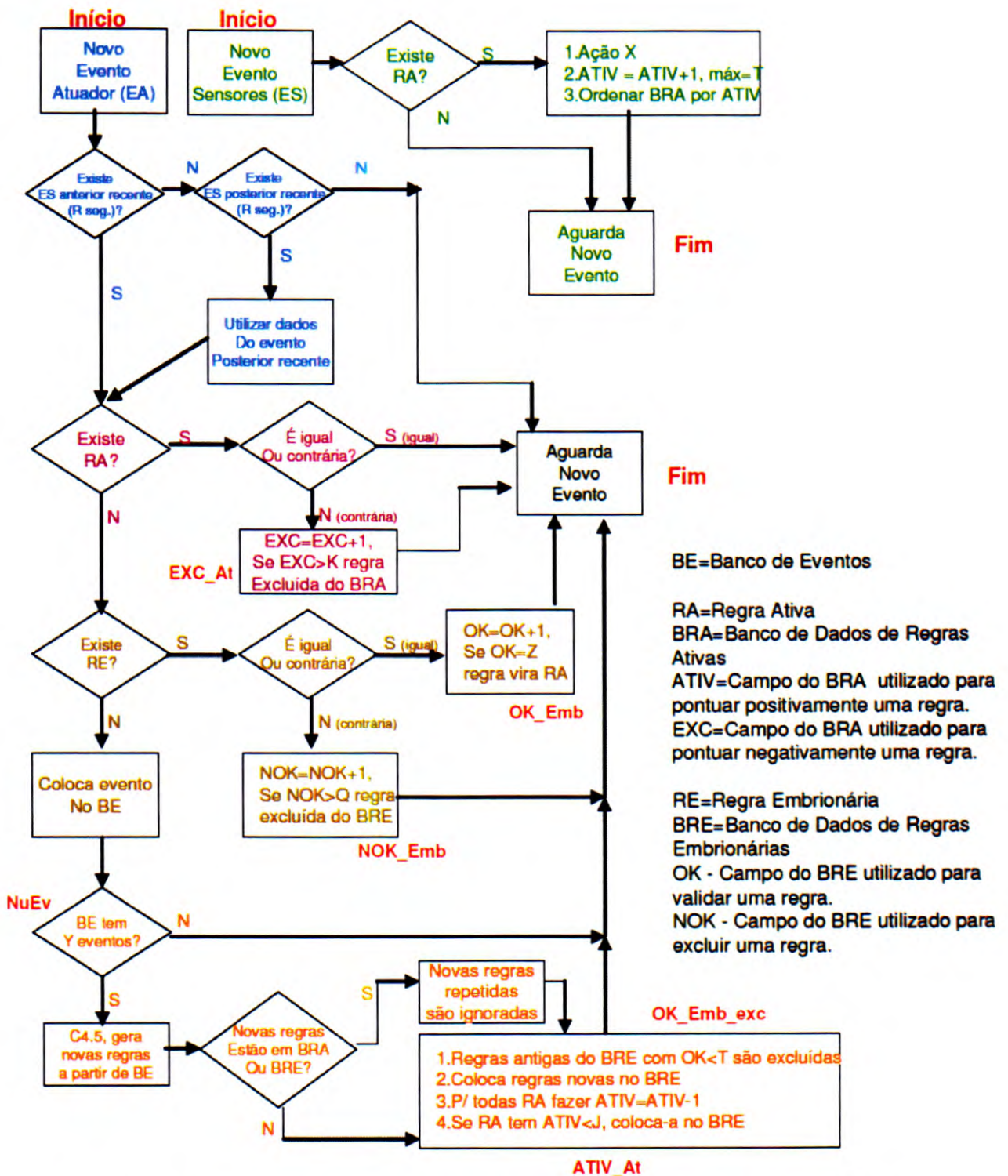


Figura 12: Fluxograma do funcionamento do Sistema ABC+. Fonte: (SGARBI, 2007).

Em seu trabalho, Sgarbi (2007) ressalta alguns pontos importantes sobre as limitações do sistema ABC+:

- O sistema aprende o que é repetitivo, isto é, a repetição de eventos é que irão validar as regras Embrionárias para que elas sejam elevadas a 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.

O autor chama a atenção ainda para o uso do sistema ABC+ em aplicações reais e mais complexas para se lidar com possíveis *loops* nas regras criadas. Um *loop* pode fazer com que uma regra ao ser ativada leve à ativação direta de outra regra ou ativação indireta de regra por mudança de algum sensor.

#### 4.3.4 Inconsistência nas Regras

A inconsistência de regras se refere a existência de regras que são contrárias entre si ou ter combinações de sensores que ativam mais do que uma regra. As providências tomadas ao se corrigir as deficiências encontradas no sistema ABC, fizeram com que se evitasse também a inconsistência de regras no sistema ABC+.

Sempre que um evento de atuador é armazenado no banco de Eventos do atuador, ele está propenso a se tornar uma Regra Embrionária. Essa Regra Embrionária será pontuada positivamente até que atinja certo valor e suba para o banco de Regras Ativas. Cada vez que um evento de sensor atender as condições dessa Regra Ativa, a mesma é executada (SGARBI, 2007).

Se uma Regra Ativa é contrariada, fazendo com que o atuador tenha um valor diferente do valor esperado (valor armazenado na regra), então ela receberá pontos negativos no seu campo EXC até que ela seja rebaixada a Regra Embrionária. Caso o habitante ainda continue repetindo a ação contrária à regra, esta Regra Embrionária receberá agora pontos negativos em seu campo NOK até que seja excluída do banco.

Se o habitante mantiver a nova rotina (a que era contrária à regra excluída), então estes eventos não irão contrariar nenhuma regra, pois a regra já foi excluída. Os eventos são então armazenados no banco de Eventos para gerarem nova Regra Embrionária e, posteriormente, Regra Ativa (SGARBI, 2007).

Assim, não existirão inconsistências já que sempre ocorrerá, primeiro, a exclusão de regras existentes antes da criação de novas regras que as possam contrariar.

## 5 IMPLEMENTAÇÃO DO ABC+

Para demonstrar o funcionamento da lógica do sistema ABC+, foi realizada a implementação do mesmo. Seguindo as definições dadas por seu autor, foi desenvolvida a Janela de Observações e os bancos de Eventos, Regras Embrionárias e Regras Ativas, que irão armazenar as regras que condizem com o comportamento do habitante, além da manutenção das mesmas.

Para o desenvolvimento do ABC+, escolheu-se a IDE (*Integrated Development Environment* - Ambiente Integrado de Desenvolvimento) Delphi 7, no qual foi colocada em forma de algoritmos a lógica do sistema. Associado ao Delphi utilizou-se o SGDB (Sistema de Gerenciamento de Banco de Dados) Firebird para guardar os Eventos e Regras Embrionárias e Ativas geradas pelo simulador. A escolha dessas ferramentas se deu pela afinidade e experiência para sua manipulação.

O ambiente para a simulação do sistema é baseado em um cômodo onde é encontrado um ventilador como atuador, que pode assumir os valores 'Ligado' e 'Desligado'. No cômodo encontram-se também quatro sensores:

- Um sensor para a porta, com os possíveis valores 'Entrada' e 'Saída';
- Um sensor de Temperatura, com os possíveis valores 'Alta', 'Normal' e 'Baixa';
- Um sensor de Luminosidade, com os possíveis valores 'Alta', 'Normal' e 'Baixa';
- Um sensor de Umidade, com os possíveis valores 'Alta', 'Média' e 'Baixa'.

As tabelas de Eventos (tab. 5), Regras Ativas (tab. 6) e Regras Embrionárias (tab. 7) do simulador foram criadas de acordo com os exemplos apresentados anteriormente nas Tabelas 2, 3 e 4. As tabelas, com algumas regras para exemplos, podem ser visualizadas a seguir:

Cod	Temperatura	Luminosidade	Umidade	Porta	Ventilador
1	Normal	Baixa	Média	Entrada	Ligado
2	Alta	Alta	Baixa	Entrada	Desligado
...					

*Tabela 5: Tabela de Eventos. Fonte: do autor.*

Cod	Temperatura	Luminosidade	Umidade	Porta	Ventilador	OK	NOK
1	Normal	Baixa	Média	Entrada	Ligado	3	1
2	Alta	Alta	Baixa	Entrada	Desligado	2	1
...							

*Tabela 6: Tabela de Regras Embrionárias. Fonte: do autor.*

Cod	Temperatura	Luminosidade	Umidade	Porta	Ventilador	ATIV	EXC
1	Normal	Baixa	Média	Entrada	Ligado	4	0
2	Alta	Alta	Baixa	Entrada	Desligado	1	1
...							

*Tabela 7: Tabela de Regras Ativas. Fonte: do autor.*

Para efeito de simulação, a execução do simulador desenvolvido distingue dois momentos: o primeiro demonstra a Janela de Observação, que representa a etapa de observação e armazenamento de eventos para que estes possam servir de entrada para o algoritmo C4.5. No segundo momento, a atenção é voltada apenas para a Manutenção das Regras Ativas e Embrionárias. Nesta etapa as regras geradas pelo C4.5 já são partes das Regras Embrionárias e não há a atuação da Janela de Observação.

As regras geradas pelo algoritmo C4.5 devem ser inseridas manualmente no simulador já que a saída produzida pelo mesmo é de um formato não comum.

A tela principal do simulador (Figura 13) possui três botões para designar cada parte implementada, sendo elas a Janela de Observação, a Manutenção das Regras e a inserção das Regras Embrionárias.





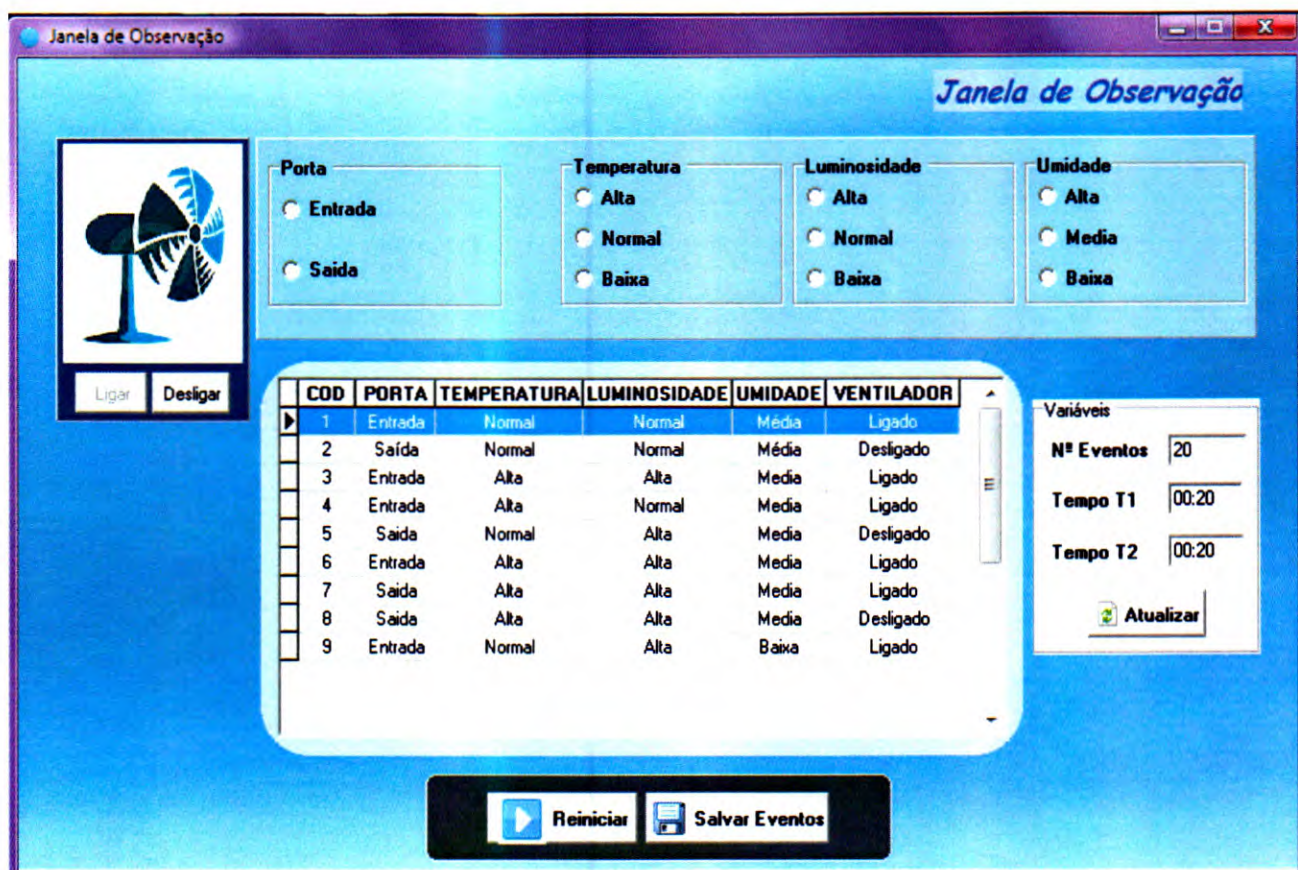
*Figura 13: Tela principal do simulador do Sistema ABC+. Fonte: do autor..*

### 5.1 Observação e armazenamento de novos eventos

Nesta primeira etapa, é demonstrada a lógica de funcionamento da Janela de Observação para identificar quando um evento de sensor está associado a um evento de atuador e, assim, gerando conjuntos de observações que serão armazenadas e inseridas no C4.5 para que este monte a árvore de decisão e posteriormente as regras que a representam.

Os elementos representantes dos sensores com seus respectivos estados foram dispostos de forma que, apenas um e somente um estado possa ser selecionado por vez, bem como os estados do atuador (Figura 14). O botão Reiniciar zera os estados dos sensores e do atuador. No *grid* (tabela) ao centro da tela podem ser visualizados os Eventos gerados.





**Figura 14:** Execução da Janela de Observação. Fonte: do autor.

O simulador fica aguardando que alguma ação seja realizada. Se uma mudança acontece em algum sensor, o horário da mudança ( $t_{Causa}$ ) é guardado e então a Janela de Observação Anterior entra em ação. Durante determinado tempo é aguardada a ocorrência de uma mudança no estado do atuador. Caso ocorra uma mudança no atuador, o seu horário ( $t_{Efeito}$ ) também é armazenado e a diferença entre estes dois horários é comparada ao tempo pré-determinado da Janela de Observação Anterior.

Caso a diferença de tempo entre os horários de alteração dos estados do sensor e atuador seja menor que o tempo da Janela de Observação ( $t_{Efeito} - t_{Causa} < T1$ ), o evento gerado é inserido automaticamente no banco de Eventos.

Se o atuador não tiver seu estado modificado ou se a diferença entre os tempos de acionamento do sensor e do atuador for maior que o tempo da Janela de Observação ( $t_{Efeito} - t_{Causa} > T1$ ), então nenhuma ação é realizada e o simulador volta ao seu estado inicial.

O tempo pré-definido adotado para a Janela de Observação, tanto para a Anterior quanto para a Posterior, foi de 20 segundos. É um espaço de tempo razoável para ser usado nesta demonstração, permitindo que o estado de outros sensores possa ser alterado com tranquilidade. A escolha desse período em uma situação real já deve levar em conta alguns



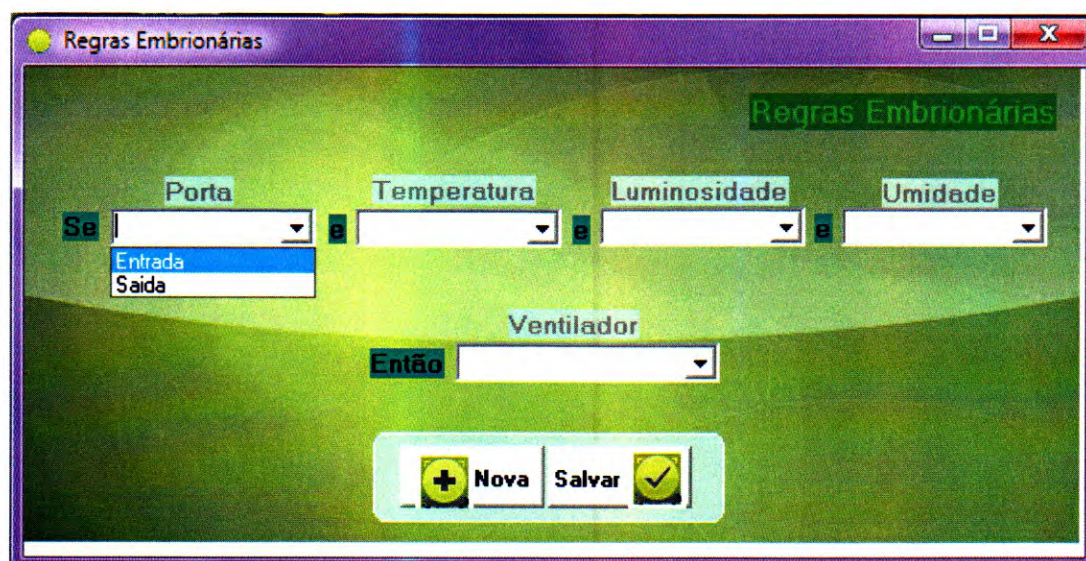
fatores como os hábitos do habitante e a estrutura e disposição dos cômodos e seus vários elementos.

A quantidade de eventos a ser inseridos no C4.5 também foi pré-determinada. O número 20 foi escolhido como padrão por ser um conjunto de amostras razoável para o algoritmo C4.5 criar as regras. Tanto os tempos da Janela de Observação Anterior (T1) e da Janela de Observação Posterior (T2), quanto à quantidade de eventos para servirem de amostra podem ser modificados no espaço Variável.

No momento em que o banco de Eventos possuir a quantidade pré-determinada de registros, esses dados serão copiados para um arquivo (Eventos.data), cujo formato faz parte das exigências do algoritmo C4.5, para que possa servir de entrada para a análise. O arquivo (Eventos.data) também poderá ser criado através do botão Salvar Eventos.

## 5.2 Inserção das Regras Embrionárias

Uma vez criada as regras pelo C4.5 a partir dos Eventos fornecidos, o algoritmo não fornece uma saída dessas regras em um arquivo (ou fornecer uma saída em formato não apropriado). Com essa deficiência, torna-se necessário que as regras geradas pelo algoritmo sejam inseridas manualmente no banco de Regras Embrionárias.



*Figura 15: Janela para inserção das Regras Embrionárias. Fonte: do autor.*

Na janela para inserção das Regras Embrionárias (Figura 15) os campos que representam os sensores possuem seus respectivos valores listados com a finalidade de tornar



esta ação mais rápida e simples. Ao ser salva, é atribuído o valor zero aos campos OK e NOK daquela regra no banco de Regras Embrionárias.

### 5.3 Manutenção das Regras Ativas e Embrionárias

Nesta etapa de Manutenção das Regras são executadas as Regras Ativas armazenadas no banco de acordo com o estado dos sensores no momento, e a validação das Regras Embrionárias para saber se podem ser elevadas ao banco das Regras Ativas ou se devem ser excluídas ao serem contrariadas frequentemente. Tal como na Janela de Observação, os estados dos sensores e do atuador são representados de forma que somente um e apenas um estado possa ser selecionador (Figura 16).

**Manutenção das Regras**

**Porta**

Entrada

Saída

**Temperatura**

Alta

Normal

Baixa

**Luminosidade**

Alta

Normal

Baixa

**Umidade**

Alta

Media

Baixa

Ligar Desligar

Label2

**REGRAS EMBRIONÁRIAS**

COD	PORTA	TEMPERATURA	LUMINOSIDADE	UMIDADE	VENTILADOR	OK	NOK
1	Entrada	Normal	Alta	Media	Ligado	1	0

**REGRAS ATIVAS**

COD	PORTA	TEMPERATURA	LUMINOSIDADE	UMIDADE	VENTILADOR	ATIV	EXC
-----	-------	-------------	--------------	---------	------------	------	-----

Variáveis

OK

NOK

ATIV

EXC

Atualizar

**Figura 16:** Janela de Manutenção das Regras Embrionárias e Ativas. Fonte: do autor.

No instante em que o estado de algum sensor é mudado, uma busca é realizada no banco de Regras Ativas, passando-se como parâmetros da busca os estados dos sensores no

momento, para saber se há alguma regra cujos estados dos sensores são iguais. Se a busca retornar um resultado, então o sistema deve realizar a mudança necessária no estado do atuador (ventilador) que consta na Regra Ativa e soma-se um ao campo ATIV da regra.

Havendo uma mudança no estado do atuador depois dele ter sido alterado pelo sistema que seguiu a instrução de uma Regra Ativa, então significa que aquela regra foi contrariada. Por isso, o valor do seu campo EXC deve ser acrescido de um.

Os campos ATIV e EXC são verificados constantemente, sempre que há uma mudança nos seus valores. Quando algum desses campos atinge a pontuação pré-determinada ou a Regra Ativa deve ser rebaixada ao banco das Regras Embrionárias, no caso do campo EXC ter atingido a pontuação, ou o campo ATIV não será mais acrescido, no caso de este campo ter atingido a pontuação. Quando o campo ATIV atinge a pontuação pré-determinada não se faz necessária uma mudança no seu valor, tendo esta regra atingida seu valor máximo.

Entretanto, se a busca por uma Regra Ativa condizente aos estados dos sensores naquele momento não retornar nenhum resultado, uma nova busca agora é realizada no banco de Regras Embrionárias.

A função das Regras Embrionárias é validar regras para que estas possam fazer parte do banco das Ativas, que são executadas pelo sistema. Porém, as Embrionárias ainda requerem mudanças no atuador realizadas por um habitante. Sendo assim, no simulador estas regras também terão o intermédio do usuário.

Quando o usuário realizar alterações nos estados dos sensores e atuador, efetua-se uma busca por uma regra Embrionária que tenha aqueles estados de sensores e atuador armazenados. Se existir tal regra, o valor do seu campo OK é incrementado em um. Mas se a busca retornar uma regra em que apenas os valores dos sensores são os mesmos daquele momento, tendo o atuador sofrido uma mudança diferente àquela armazenada na regra, então essa regra foi contradita e seu valor NOK deve ser somado a um.

Os campos OK e NOK das Regras Embrionárias também possuem valores pré-definidos. O campo OK ao atingir a pontuação máxima designa que aquela regra foi validade e pode passar a fazer parte do banco das Ativas. Ela é então excluída do banco das Embrionárias e inserida no banco das Ativas.

Por outro lado, se o campo NOK atingir seu valor máximo é determinado que aquela regra não seja válida e é então removida do banco das Embrionárias. Com essas medidas de controle das Regras Ativas e Embrionárias é possível eliminar inconsistências durante a execução do sistema ABC+.

Os valores dos campos OK, NOK, ATIV e EXC podem ser modificados no espaço das Variáveis. Para exemplificação, adotou-se o valor 6 para os campos OK e ATIV e o valor 4 para os campos NOK e EXC.

A implementação do simulador para o sistema ABC+ foi codificado de acordo com o que foi descrito por Sgarbi (2007) em relação à Janela de Observações e Manutenção das regras Embrionárias e Ativas.

## 6 CONCLUSÃO E TRABALHOS FUTUROS

### 6.1 Considerações Finais

Neste trabalho foi abordado o tema Domótica Inteligente, um assunto até então recente e com poucas explorações. Foi proposta a simulação de um sistema domótico, o ABC+, que integrado ao algoritmo C4.5 generaliza e cria regras sobre o comportamento do habitante em uma casa. Essas regras se baseiam em um conjunto de observações apresentadas por sensores e atuador presentes no ambiente.

O trabalho realizado tem por objetivo introduzir e incentivar o estudo na área de Domótica Inteligente. Através do exemplo do sistema de automação residencial desenvolvido pode-se mostrar que é possível adaptar um ambiente às preferências de seus usuários através da utilização dos conceitos de Domótica e Aprendizado Automático. Essas adaptações acompanham as mudanças de hábitos por meio do aprendizado automático, um ramo da Inteligência Artificial.

O simulador do sistema ABC+ foi capaz, através da lógica presente na Janela de Observação, de interpretar os eventos realizados, eliminando aqueles eventos que não possuíam vínculos entre si. Por meio das Regras Embrionárias podem-se validar as regras que realmente condizem com as preferências dos habitantes da casa. E a Manutenção das Regras, como o esperado, executou o seu papel de rebaixar as Regras Ativas que não estavam mais de acordo com os hábitos do habitante e promovendo as Regras Embrionárias que provaram ser válidas.

As pesquisas realizadas para o desenvolvimento desse trabalho somaram conhecimentos a cerca do Aprendizado Automático e suas diversas abordagens para a aquisição de conhecimento; da Busca por Conhecimento em Banco de Dados que se mostra útil e eficaz através da utilização do algoritmo C4.5 para indução de regras de decisão; e finalmente sobre Domótica Inteligente e sua ampla possibilidade a serem exploradas.

### 6.2 Trabalhos Futuros

Com a implementação do simulador observou-se alguns pontos que devem ser melhor analisados para um melhor funcionamento do sistema ABC+, mas que serão abordados em trabalhos futuros.

Um desses pontos são os valores que devem ser atribuídos às variáveis nEventos, OK, NOK, ATIV, EXC, T1 e T2. Esses valores devem ser analisados futuramente para um melhor desempenho do sistema em aplicações reais, bem como a quantidade de sensores que devem ser utilizados para uma melhor análise das situações geradas.

Outro ponto é a deficiência do sistema ABC+ em trabalhar apenas com um habitante por não ser capaz de diferenciar um habitante de outro. Como solução para isso, poderia ser utilizada a tecnologia RFID (*Radio Frequency Identification*), que são sistemas automáticos de identificação que utilizam sinais de rádio frequência para identificar e localizar etiquetas eletrônicas, onde são armazenados dados, e que podem ser aderidas em objetos. Ou ainda a utilização de reconhecimento de voz, em que cada habitante poderia ser reconhecido ao falar algo dentro de um cômodo.

E ainda, soluções para possíveis *loops* entre as regras devem ser buscadas. O *loop* pode levar ao travamento do sistema, má execução das regras e problemas nos dispositivos da residência. Assim, é importante prever e eliminar esses *loops*.



## REFERÊNCIAS BIBLIOGRAFICAS

ALVES, José Augusto; MOTA, José. (2003). **Casas Inteligentes**. Editora Centro Atlântico Ltda. Portugal, 2003. 144p.

ARAÚJO, Regina Borges. **Computação Ubíqua: Princípios, Tecnologias e Desafios**. São Carlos-SP, 2003.

BATISTA, G. E. **Pré-processamento de Dados em Aprendizado de Máquina**. São Carlos, 2003.

BERNARDES, Ricardo Martins. **C4.5: Um recurso para geração de Árvores de Decisão**. Campinas-SP, 2001.

BOLZANI, Caio Augusto Morais. **Análises de Arquiteturas e Desenvolvimento de uma Plataforma para Residências Inteligentes**. São Paulo, 2010.

BRESSLER, Fábio. **Um protótipo de aplicação para recomendação de produtos baseado no interesse e comportamento de uso do usuário**. São Leopoldo, 2004. Disponível em: <[http://www.unisinos.br/inf/images/stories/Inf/24tc\\_fabio\\_bressler.pdf](http://www.unisinos.br/inf/images/stories/Inf/24tc_fabio_bressler.pdf)>. Acessado em 03 out. 2010.

BRETERNITZ, J. V. **Domótica: as casas inteligentes**. 2001. apud SGARBI, J A., **Domótica Inteligente: Automação Baseada Em Comportamento**. São Bernardo do Campo, 2007.

COSTA, Welbson Siqueira. **Extração de Regras Simbólicas de Agrupamento de Dados de Expressão Gênica**. Natal, 2006.

**Dicionário Priberam da Língua Portuguesa**, 2010. Disponível em: <<http://www.priberam.pt/DLPO/default.aspx?pal=intelig%C3%Aancia>>. Acessado em: 15 out. 2010.

FAYYAD, Usama. et al. **From Data Mining to Knowledge Discovery in Databases**. 1996. Disponível em: <http://www.kdnuggets.com/gpspubs/aimag-kdd-overview-1996-Fayyad.pdf>> Acessado em 10 jan. 2011.

FONSECA, J. M. M. R. **Indução de Árvores de Decisão: HistClass – Proposta de um algoritmo não paramétrico**. Lisboa, 1994.

FREITAS, Alex A. **Data Mining**, In: XIII Simpósio Brasileiro de Banco de Dados. Maringá, 1998. apud JOST, Ingo. **Mineração de Dados para Adoção de Práticas de Marketing em Ambiente Acadêmico**. Novo Hamburgo, 2009

GOLDSCHMIDT, Ronaldo; PASSOS, Emmanuel. **Data Mining : Um guia prático**. Rio de Janeiro: Elsevier, 2005.

GRÉGIO, André Ricardo Abed. **Aplicação das Técnicas de Data Mining para a Análise de Logs de Tráfego TCP/IP**. São José dos Campos, 2007. Disponível em: <



[m17.sid.inpe.br/col/sid.inpe.br/mtc-m17@80/2007/04.04.18.29/doc/publicacao.pdf](http://m17.sid.inpe.br/col/sid.inpe.br/mtc-m17@80/2007/04.04.18.29/doc/publicacao.pdf)>. Acessado em 12 jan. 2011.

JOST, Ingo. **Mineração de Dados para Adoção de Práticas de Marketing em Ambiente Acadêmico**. Novo Hamburgo, 2009. Disponível em: <[http://tconline.feevale.br/tc/files/0001\\_1931.pdf](http://tconline.feevale.br/tc/files/0001_1931.pdf)>. Acessado em 13 out. 2010.

KOERICH, Alessandro L. **Aprendizado de Máquina – Introdução**. 2008 Disponível em: <<http://www.ppgia.pucpr.br/~alekoe/AM/2008/1-Introducao-ApreMaq-2008.pdf>>. Acessado em: 20 out. 2010.

LAROUSSE. **Grande Enciclopédia Larousse Cultural**. Editora Nova Cultural, 1999.

LIU, Bing et al. **Clustering Through Decision Tree Construction**. 2000. apud COSTA, Welbson Siqueira. **Extração de Regras Simbólicas de Agrupamento de Dados de Expressão Gênica**. Natal, 2006.

MATOS, Paulo Freire et al. **Relatório Técnico “Conceitos sobre Aprendizado de Máquina”**. São Carlos, 2009.

MICHALSKI, R.S.; BRATKO, I.; KUBAT M. **Machine Learning and Data Mining. Methods and Applications**. Wiley & Sons Ltd., Estados Unidos, 1998. apud SGARBI, J A., **Domótica Inteligente: Automação Baseada Em Comportamento**. São Bernardo do Campo, 2007.

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

MONARD, Maria Carolina. et al. **Uma Introdução ao Aprendizado Simbólico de Máquina por Exemplos**. 1997

MONARD, Maria Carolina; PRATI, Ronaldo Cristiano. **Aprendizado de Máquina Simbólico para Mineração de Dados**. São Carlos. 2005.

NAVEGA, Sergio. **Princípios Essenciais do Data Mining**. Publicado nos Anais do Infoimagem, Cenadem, São Paulo, 2002. Disponível em: <http://www.intelliwise.com/reports/i2002.pdf> . Acessado em 16 nov. 2010.

NIKOLOPOULOS, Chris. **Expert Systems – Introduction to First and Second Generation and Hybrid Knowledge Based Systems**. Marcel Dekker Inc. Press. 1997. apud OSORIO, F. S. **Tutorial: Redes Neurais – Aprendizado Artificial**, 2010.

OSORIO, F. S. **Tutorial: Redes Neurais – Aprendizado Artificial**. Disponível em: <<http://osorio.wait4.org/oldsite/IForumIA/fia99.pdf>. 1999>. Acessado em 30 set. 2010.

POZO, Aurora Trinidad Ramirez. **Tutorial de Árvores de Decisão**. Disponível em: <<http://www.inf.ufpr.br/aurora/tutoriais/arvoresdecisao/>>. Acessado em 15 set. 2010.

QUILAN, J. Ross. **C4.5: Programs for Machine Learning**. Morgan Kaufmann Publishers. San Mateo, CA. 1993.

QUILAN, J. Ross. **Is See5/C5.0 Better Than C4.5?** 2009. Disponível em: <<http://www.rulequest.com/see5-comparison.html>>. Acessado em 16 fev. 2011.

REZENDE, Solange Oliveira. **Mineração de Dados**. São Leopoldo, 2005. Disponível em: <<http://bibliotecadigital.sbc.org.br/download.php?paper=417>>. Acessado em 30 set. 2010.

RUSSELL, S. J.; NORVIG, P. **Inteligência Artificial**. Editora Campus, Rio de Janeiro. 2004.

SERVENTE, Magdalena. **Algoritmos TDIDT Aplicados a la Minería de Datos Inteligente**. Buenos Aires, 2002.

SGARBI, J. A., TONIDADEL, F. **Aprendendo Regras por Observação em uma Casa Inteligente**. Congresso Brasileiro de Automática, 2006a.

SGARBI, Julio A., TONIDANDEL, Flávio. **Domótica Inteligente: Automação Residencial baseada em Comportamento**. In: Workshop de Teses e Dissertação em IA. International Joint Conference IBERAMIA/SBIA.2006b.

SGARBI, J A., **Domótica Inteligente: Automação Baseada Em Comportamento**. São Bernardo do Campo, 2007.

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

TURING, A.M. **Computing machinery and intelligence**. 1950. Disponível em: <<http://www.loebner.net/Prizef/TuringArticle.html>>. Acessado em: 26 out. 2010.

WINSTON, P. H. **Artificial Intelligence**. Addison-Wesley. 1984.

ZADROZNY, Bianca. **Aprendizado de máquina, Aula 1**. 2010. Disponível em: <<http://www.ic.uff.br/~bianca/aa/>>. Acessado em 13 out. 2010.

**ANEXOS**

## ANEXO A

## ALGORITMOS PARA EVENTO DE SENSOR E EVENTO DE ATUADOR

## 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

## 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

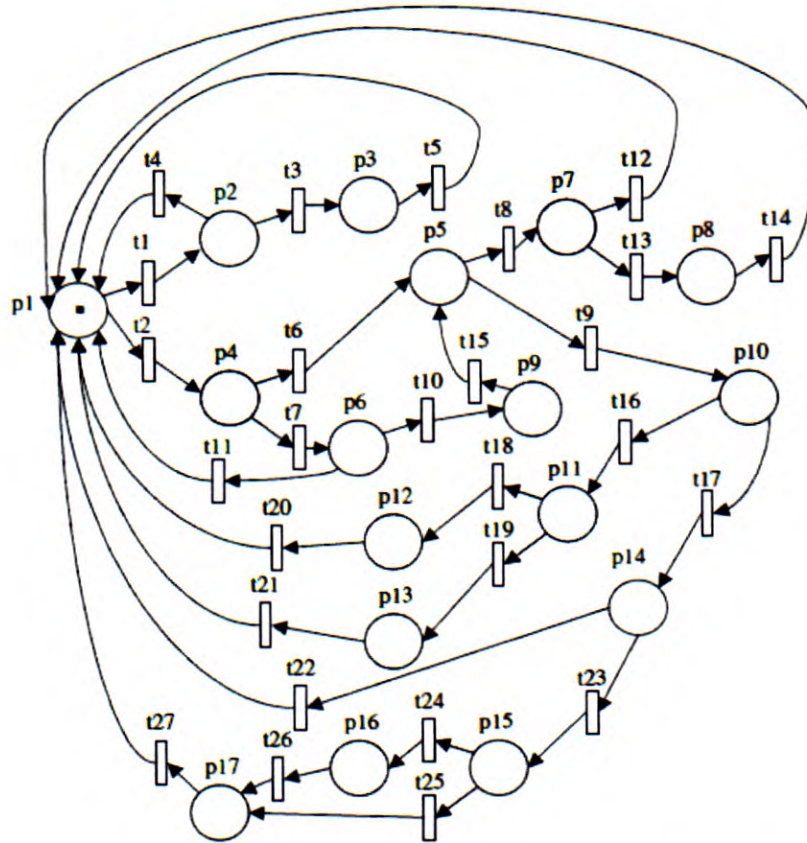
Colocar o evento no BDEventos;  
 Se BDEventos igual a NuEv,  
     Acionar C4.5 para gerar novas regras;  
     Se novas regras já estão no BDAtivas ou BDEmbrio,  
         Ignorar as regras repetidas;  
     Se alguma Regra Embrionária tem campo OK menor que OK\_Emb\_exc,  
         Excluir Regra Embrionária;  
     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



## ANEXO B

## REDE DE PETRI DO SISTEMA ABC+



## 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  $ATTV=ATTV+1$ , com  $ATTV$  máximo= $T$  e ordenação do BRA por  $ATTV$   
 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  $ATTV=ATTV-1$ ; se RA tem  $ATTV<ATTV\_At$  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 – Algumas(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