Implementação de Redes Bayesianas em Sistemas Embarcados

Fabio Tozeto Ramos *   Fabiana Mikami   Fabio Gagliardi Cozman†

Escola Politécnica, Universidade de São Paulo
Av. Prof. Mello Moraes, 2231, 05508-900, São Paulo, SP - Brazil
fabioram@usp.br, fabimikami@hotmail.com, fgcozman@usp.br

Abstract. Este trabalho descreve um sistema de inferência probabilístico adaptado à plataforma Palm de processamento móvel. O objetivo do trabalho é realizar análise de falhas em equipamentos ou processos de produção. O sistema aqui descrito foi instalado em um PalmIIIe com 128KB de memória dinâmica e 2048KB de memória estática. O Palm tem capacidade de se conectar a um Controlador Lógico Programável e capturar o estado de variáveis analógicas e digitais em uma planta industrial. Com base nessas observações, o sistema de inferências embarcado no Palm pode gerar probabilidades de ocorrência de falhas.

1 Introdução

No momento, sistemas de processamento embarcado e sistemas de processamento móvel passam por uma revolução. Com a popularização dos telefones celulares, agendas eletrônicas de bolso e mesmo brinquedos interativos, sistemas embarcados microprocessados crescem em flexibilidade e autonomia. Em ambiente industrial, sistemas embarcados de alta miniaturização atuam em programas de reconhecimento de padrões e manipulação de bancos de dados. Ao mesmo tempo, o avanço das tecnologias de redes e de transmissão de dados têm levado ao aparecimento de vários dispositivos para processamento móvel ou remoto.

Atualmente, uma das plataformas mais populares para computação móvel é o Palm, originalmente estruturado como um assistente pessoal. Devido à grande capacidade de armazenamento e ao tamanho reduzido, um Palm está habilitado a realizar processamento relativamente complexo de forma embarcada ou móvel.

O objetivo deste trabalho é explorar a plataforma Palm em aplicações que envolvem raciocínio probabilístico; mais precisamente, aplicações que envolvem análise estatística de falhas. Para tanto, foi desenvolvido um sistema de inferência adequado para instalação em um Palm. O sistema de inferências processa redes Bayesianas com número qualquer de variáveis categóricas (Seção 2).

Programas que contém raciocínio baseado em redes Bayesianas são usualmente implementados sem maiores preocupações em relação a consumo de memória. Normalmente são construídos de forma monolítica, ocupando vastas quantidades de recursos. Entretanto, aplicações em sistemas embarcados requerem uma abordagem diferente e geram novos problemas teóricos e práticos. Este trabalho é um primeiro passo na pesquisa de algoritmos e técnicas probabilísticas com baixo consumo de recursos computacionais. A implementação de um sistema de redes Bayesianas em um Palm deve ser visto como uma primeira etapa; o objetivo maior é a criação de uma infraestrutura de processamento remoto/móvel/embarcado adequada a aplicações industriais e comerciais.

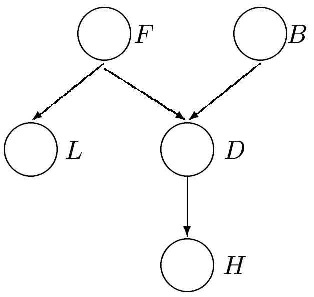
No trabalho realizado, os algoritmos de inferência do sistema JavaBayes, um programa para manipulação, processamento e inferência de redes Bayesianas, foram transferidos para a máquina virtual Waba e testadas em um Palm IIIe. O programa resultante foi intitulado PalmBayes. O sistema PalmBayes ocupa menos de 100KBytes de memória e utiliza memória de forma otimizada. O Palm IIIe deverá ser conectado a um equipamento típico de automação, como um CLP (computador lógico programável), a fim de realizar análise de falhas de processos de produção. Por exemplo, análise de uma planta química ou das características de operação de um motor de indução. A Seção 3 descreve o ambiente de trabalho utilizado no trabalho. A Seção 4 descreve o programa PalmBayes e a Seção 5 apresenta aplicações do conceito.

2 Redes Bayesianas

Modelos estatísticos com grande número de variáveis oferecem desafios computacionais imensos para sistemas de suporte a decisões. Existem várias estruturas gráficas, como as redes Bayesianas, que visam facilitar a representação e a manipulação de distribuições de probabilidade multivaríaveis.

*Bolsista do Programa Especial de Treinamento PET, da CAPES.

†Parcialmente financiado pelo CNPq no processo 300183/98-4.



F: family-out
L: light-on
D: dog-out
B: bowel-problem
H: hear-bark

Probabilidades:
$p(F)$, $p(B)$, $p(L|F)$, $p(D|F,B)$, $p(H|D)$

Figura 1: Exemplo de rede Bayesiana ("DogProblem").

Redes Bayesianas são grafos direcionados que codificam distribuições de probabilidade [8, 9]. Modelos baseados em redes Bayesianas têm sido utilizados para sistemas de suporte a decisões médicas, sistemas de auxílio ao usuário, sistemas de monitoração e processamento de dados estatísticos e informações de processos industriais. Algoritmos para redes Bayesianas se beneficiam da estrutura gráfica da rede para cancelar cálculos e simplificar operações.

Uma rede Bayesiana especifica uma distribuição conjunta sobre variáveis X_i através de um grafo direcionado. Denote por $\mathrm{pa}(X_i)$ o conjunto de variáveis que são os pais diretos da variável X_i no grafo. Cada variável é associada a uma distribuição $p(Xi|\mathrm{pa}(Xi))$.

A convenção básica para construção de uma rede Bayesiana é que toda variável X_i é independente de todos os não-descendentes de X_i, dados os pais de X_i. Essa propriedade garante que a distribuição conjunta representada por uma rede Bayesiana seja definida por:

$$p(X_1, \ldots, X_n) = \prod_i p(X_i|pa(X_i)).$$

A expressão genérica para o cálculo da distribuição condicional de uma variável qualquer X_q, dadas observações E, é:

$$p(X_q|E) = \frac{\sum_{X_i \notin \{X_q, E\}} \prod_i p(X_i|pa(X_i))}{\sum_{X_i \notin \{E\}} \prod_i p(X_i|pa(X_i))}. \qquad (1)$$

Um exemplo de redes Bayesianas é mostrado na Figura 1, conforme descrito por Charniak [1].

Aplicações em sistemas embarcados exigem algoritmos com baixo consumo de memória. No momento existem poucos algoritmos na literatura que levam consumo de memória em conta. O problema teórico pode ser entendido da seguinte forma. Tomando a Expressão (1), pode-se ver que um algoritmo de inferência deve calcular a seguinte somatória eficientemente:

$$\sum_{X_i \notin \{X_q, E\}} \prod_i p(X_i|pa(X_i)),$$

visto que o denominador da Expressão (1) é apenas uma constante de proporcionalidade. Para calcular esta expressão, deve-se reorganizar os termos, intercalando produtos e somas de forma eficiente [2, 6]. Por exemplo, o cálculo de $p(L, H = h)$, tomando a rede Bayesiana da Figura 1, para um valor fixo de H, é:

$$\sum_F p(F)p(L|F) \sum_B p(B) \sum_D p(D|F,B)p(H = h|D).$$

Em uma rede Bayesiana com grande número de variáveis, os resultados intermediários de somatórias podem consumir quantidade excessiva de memória. Pode-se diminuir o consumo de memória diminuindo o tamanho dos resultados intermediários. Uma idéia é realizar várias somatórias ao mesmo tempo, sem armazenar resultados intermediários — Dechter usa o termo "super-bucket" para se referir aos termos resultantes nessa abordagem [5]. Uma segunda idéia é selecionar algumas variáveis como "pivôs" e fixá-las durante somatórios. Por exemplo, suponha que fixemos a variável binária B na expressão anterior. Devemos então realizar os produtos e somas duas vezes, uma para cada valor de B. Note que o tempo de processamento essencialmente dobrou, enquanto o consumo de memória diminuiu — visto que não é necessário armazenar valores de B em resultados intermediários [6]. Uma abordagem significativamente diferente é modificar a forma de armazenamento de dados em redes Bayesianas, criando estruturas de dados intermediárias que armazenam computações semi-prontas. Essa abordagem tem sido explorada por Darwiche [4] e Provam [3]. Até o momento, esses esforços iniciais no sentido de reduzir consumo de memória ainda não foram comparados e sistematizados. De fato, a maioria desses esforços tem sido teóricos, sequer levando a implementações reais.

Caso não haja memória disponível para realizar produtos e somas, o que é possível calcular? Nesse caso, ainda é possível recorrer a métodos aproximados. Métodos de Monte Carlo, como por exemplo Gibbs sampling, requerem pouca memória e admitem implementações bastante simples [7]. Portanto pode-se imaginar um espectro amplo de variantes, combinando:

1) Algoritmos exatos e unificados

2) Algoritmos exatos em etapas/super-buckets

3) Algoritmos aproximados de Monte Carlo

No momento não existem técnicas que permitam uma transição suave entre esses algoritmos, nem existem estudos teóricos ou empíricos que possam orientar a escolha de algoritmos. É nessa direção que nossa pesquisa atual se desenvolve.

3 A plataforma Palm

A plataforma Palm pode ser encontrada em uma variedade de implementações, sempre destinadas a permitir computação móvel em formato reduzido. Os comandos do usuário são passados através de uma tela sensível onde se pode escrever letras e números. Sistemas que implementam a plataforma Palm também podem se comunicar com um computador de mesa por interface serial.

Um aparelho Palm IIIe foi escolhido para testar o programa PalmBayes. Um Palm IIIe possui o processador Motorola 68328 "Dragonball" CPU, que tem 16 Mhz clock e executa aproximadamente 2.7 MIPS (millions of instructions per second). O modelo utilizado, um PalmIIIe, tem 2 MB de memória estática para armazenar programas e cerca de 128KB de memória dinâmica (heap) para as variáveis e objetos criados quando o programa é executado. Comparando-se a performance do Palm e a de um Pentium 233 Mhz, no processamento de um programa de 32 bits, nota-se que o Palm é cerca de 300 vezes mais lento.

Existem duas implementações da linguagem Java para o Palm IIIe. A Sun Microsystems lançou em 1999 uma "Java virtual machine" para o Palm IIIe, denominada KVM.[1] Independentemente, a WabaSoft desenvolveu uma máquina virtual para sistemas operacionais embarcados como o PalmOS e WindowsCE. Tais sistemas são utilizados em PDAs, minicomputadores e interfaces homem-máquina, apenas para citar alguns exemplos.

Para implementar o programa, a máquina virtual escolhida foi a Waba (da Wabasoft), pois esta oferece alguns recursos que consideramos essenciais para o desenvolvimento do PalmBayes, como: manipulação de variáveis tipo `float`, classes para comunicação serial e uso de código nativo.

De maneira geral um programa para a máquina virtual Waba consiste em um conjunto de classes que ficam "empacotadas" em um único arquivo tipo .prc. Para esse programa o Waba permite alocar os quatro tipos de memória: object heap, class heap, stack e native stack (dynamic memory) e class heap (storage

[1] A "Java virtual machine" é um programa que permite que programas feitos em linguagem Java sejam compatíveis com diferentes equipamentos e sistemas operacionais. A "virtual machine" encontra-se entre o programa e a plataforma de implementação, interpretando o programa em um código de máquina apropriado para o hardware e sistema operacional utilizado. Além disso, ela gerencia a memória do sistema e tarefas múltiplas da execução do programa.

Figura 2: Tela inicial do sistema no emulador Palm.

memory). Essas quantidades influem decisivamente no funcionamento do programa. Os testes realizados mostraram que, dependendo da complexidade do programa, apenas a Waba pode executá-lo de forma satisfatória através de um gerenciamento de memória adequado.

4 Implementação

A primeira etapa no processo de implementação do PalmBayes com o Waba, foi o desenvolvimento de uma interface com o usuário que fosse ao mesmo tempo simples e fácil de operar e não sobrecarregasse os recursos do sistema. Para isso foi utilizado o conjunto de APIs do Waba no desenvolvimento de duas classes que implementam uma tela de apresentação e a tela principal. A classe que implementa a tela principal é responsável por declarar e inicializar todos os tipos básicos da interface principal. Esta classe foi projetada seguindo o padrão do pacote EmBayes, parte do programa JavaBayes. A Figura 2 mostra a tela inicial do sistema.

O programa JavaBayes permite a construção e visualização de redes Bayesianas, bem como o cálculo de probabilidades a partir de redes Bayesianas. O sistema está disponível livremente no endereço http://www.cs.cmu.edu/~javabayes. Três partes principais compõe o sistema: interface gráfica, interpretadores de arquivos de entrada e saída, e motor de inferências. O código que implementa o motor de inferências foi agrupado em um pacote denominado EmBayes. É esse pacote que fornece a funcionalidade básica de inferências utilizada no PalmBayes.

O motor de inferências implementado no pacote EmBayes realiza as operações descritas pela Expressão (1) utilizando o algoritmo de *eliminação de variáveis* [2, 6]. Além de probabilidades marginais, o motor de inferências também calcula configurações de probabilidade máxima e realiza análise de conectividade em redes Bayesianas. O pacote EmBayes foi adaptado para utilização no Palm IIIe. A primeira etapa foi a substituição de todos os tipos primitivos `double` por `float`, visto que não há suporte para tipo `double` no Palm. Em seguida, foi feita uma análise de todas as APIs utilizadas pelo EmBayes e as disponíveis no pacote do Waba, a fim de avaliar as diferenças. A Waba possui bibliotecas reduzidas e nem sempre compatíveis com a especificação padrão da linguagem Java. Algumas classes ausentes em Waba foram adaptadas ou codificadas especialmente para o programa PalmBayes: Vector e Enumeration (manipulação de vetores) e Stack (Manipulação de pilhas).

A classe PalmBayes controla todo o programa embarcado, interpretando os comandos passados pelo usuário e acionando as classes responsáveis do pacote EmBayes. Além disso, a classe PalmBayes inicializa as classes que fazem a interface e controla os eventos associados aos elementos da interface como botões e campos de entrada.

5 Aplicações

Nesta sessão, procurar-se-á dar uma idéia de possíveis aplicações para redes Bayesianas embarcadas ou móveis.

Uma possibilidade é a utilização do PalmBayes juntamente com um microCLP conectado a uma rede industrial. A Figura 3 ilustra essa idéia. Nesse caso, o PalmBayes trocaria informações com o CLP, obtendo o estado da rede e fazendo um diagnóstico de falhas. Para o CLP em estudo no momento, um S7-200 da Siemens, a comunicação se daria através da porta serial do Palm com um conversor RS-232/485. Quando necessário, o usuário plugaria o Palm no CLP, já com um programa próprio para ser utilizado juntamente com o Palm, e com ele faria a análise de uma máquina ou do processo de produção da indústria. Esse sistema poderia ser utilizado, por exemplo, juntamente com um motor de indução trifásico onde, através da coleta de informações como corrente na armadura, tensão no estator e fluxo magnético, daria um diagnóstico da vida útil do motor ou previsibilidade de falhas. Outra alternativa é sua utilização em aparelhos de cromatografia, muito comuns em indústrias químicas. Esses aparelhos têm por função obter a quantidade mássica de determinadas substâncias químicas em uma amostra, a partir da comparação com padrões pré-definidos. Todo o processo é feito por espectometria e inevitavelmente ocorrem erros seja devido ao desgaste de algumas peças, ou contaminações por uso incorreto do operador. Neste caso, o Palm se conectaria a essa máquina através do CLP, ou diretamente a partir de alguma porta serial existente, a fim de fazer suas observações. Com isso ele pode indicar as causas mais prováveis das falhas do equipamento.

Figura 3: Palm conectado a microCLP.

Como exemplo de uma aplicação em sistema embarcado, deixando a plataforma Palm, considere a interface homem-máquina conhecida comercialmente por HMI. Essas HMIs são telas touch-screen ou terminais conectados à rede de automação de uma indústria. Dentre as redes mais conhecidas estão a PROFIBUS-DP e a DEVICE NET. O sistema operacional utilizado por eles é o WindowsCE, onde o Waba pode ser instalado. Assim, para essa aplicação, não seria utilizado o Palm mas a plataforma desses sistemas que apresentam maior capacidade de memória e desempenho. A função do sistema embarcado nesse caso, seria a de obter o estado do sistema de automação através da rede, e fazer um diagnóstico das falhas desse sistema.

Outra aplicação é a utilização do sistema para fazer a análise de todo um processo. Por exemplo, o processo de produção de cerveja envolve diversas etapas como maturação da cevada, fermentação, engarrafamento e outros. A partir da análise do produto final, ou seja, da cerveja engarrafada, um técnico de chão-de-fábrica pode, com o auxílio do Palm, verificar a causa de possíveis alterações no produto final. Essas falhas podem ser desde a má qualidade da cevada, até problemas no controle da temperatura do processo de fermentação.

6 Conclusão

Devido às restrições de processamento e memória do Palm, foram feitas várias pesquisas sobre a tecnologia mais adequada no desenvolvimento do projeto e buscou-se sempre otimizar o programa. O resultado alcançado foi satisfatório, pois o algoritmo do Palm quando executado com a mesma rede e mesmos comandos do JavaBayes, produz as mesmas inferências (apesar do tempo de processamento ser maior). O próximo passo é derivar e implementar algoritmos de inferência que levem em conta o consumo de memória.

Como atividades futuras, pretende-se que o PalmBayes seja utilizado com uma das aplicações propostas. Para isso serão desenvolvidas interfaces do programa específicas para cada aplicação, onde técnicos, mesmo sem ter conhecimento de redes Bayesianas, possam fazer suas análises.

Referências

[1] E. Charniak. Bayesian networks without tears. *AI Magazine*, pages 50–63, Fall 1991.

[2] F. G. Cozman. Generalizing variable elimination in Bayesian networks. To appear, *Workshop on Probabilistic Reasoning in Artificial Intelligence at IBERAMIA*, Atibaia, São Paulo, Brazil, 2000.

[3] A. Darviche and G. Provan. Query DAGs: A practical paradigm for implementing belief-network inference. In E. Horvitz and F. Jensen, editors, *Proceedings of the Twelfth Conference on Uncertainty in Artificial Intelligence*, pages 203–210, San Francisco, California, United States, 1996. Morgan Kaufmann.

[4] A. Darwiche. Any-space probabilistic inference. In C. Boutilier and M. Goldszmidt, editors, *Proceedings of the Sixteenth Conference on Uncertainty in Artificial Intelligence*, pages 133–142, San Francisco, California, United States, 2000. Morgan Kaufmann.

[5] R. Dechter. Topological parameters for time-space tradeoff. In E. Horvitz and F. Jensen, editors, *Proceedings of the Twelfth Conference on Uncertainty in Artificial Intelligence*, pages 220–227, San Francisco, California, United States, 1996. Morgan Kaufmann.

[6] R. Dechter. Bucket elimination: A unifying framework for probabilistic inference. In M. I. Jordan, editor, *Learning in Graphical Models*, pages 75–104. MIT Press, Dordecht, The Netherlands, 1999.

[7] W. R. Gilks, S. Richardson and D. J. Spiegelhalter. *Markov Chain Monte Carlo in Practice*. Chapman and Hall, London, England, 1996.

[8] F. V. Jensen. *An Introduction to Bayesian Networks*. Springer Verlag, New York, 1996.

[9] J. Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann, San Mateo, California, 1988.