

Paralelização do algoritmo SPRINT usando MyGrid

Juliana Carvalho, Ricardo Rebouças e Vasco Furtado
Universidade de Fortaleza – UNIFOR

juliana@edu.unifor.br
ricardo@sspds.ce.gov.br
vasco@unifor.br

1. Introdução

Mineração de Dados é o processo de analisar bases de dados em busca de padrões válidos. Atualmente, o volume das bases de dados está ficando cada vez maior e a análise destes dados cada vez mais difícil e demorada. As aplicações de Mineração de Dados com seus algoritmos para identificar padrões significativos em bases de dados volumosas são apresentadas como uma solução para esse problema. Entretanto, o processamento de algoritmos de Mineração de Dados em grandes bases de dados demanda grande capacidade de processamento. As ferramentas de Mineração de Dados necessitam de sistemas de alta performance para que bases de dados extensas possam ser “varridas” rapidamente e usuários possam compreender dados complexos, fazendo predicções com acurácia.

Uma alternativa para a demanda de alto poder de processamento das máquinas é a utilização grids computacionais. Em grids, o poder de processamento dos computadores em uma rede é utilizado para facilitar a execução de tarefas que seriam executadas em uma única máquina. As tarefas a serem executadas são divididas e distribuídas entre os computadores da rede, agilizando assim a execução das mesmas.

Neste trabalho descreveremos o algoritmo SPRINT, um algoritmo classificador baseado em árvores de decisão para Mineração de Dados, e sua paralelização em um *grid* desenvolvido pela Universidade Federal de Campina Grande - UFCG, chamado MyGrid. O algoritmo SPRINT é ideal para Mineração de Dados, pois pode analisar grandes bases de dados sem se preocupar com restrição de memória. A paralelização do algoritmo SPRINT em MyGrid também resolve o problema de demanda de alto poder de processamento e da disponibilidade de recursos.

2. Background Knowledge

2.1 MyGrid

Um *grid* é um software que permite acessar e gerenciar recursos de hardware e software distribuídos em uma rede.

Diversos *grids* estão sendo desenvolvidos, mas muitos desafios ainda necessitam ser vencidos para que os mesmos possam ser amplamente utilizados como plataformas de execução de alto desempenho para aplicações paralelas [4] [5]. Um *grid* que se presta a executar tarefas *Bag-of-Tasks* (BoT), ou seja, tarefas paralelas e independentes, e assim estar mais rapidamente disponível aos usuários é o MyGrid.

A tecnologia MyGrid trabalha com aplicações BoT. Um *grid* é composto por uma máquina chamada de *Home Machine* e várias máquinas chamadas de *Grid Machines*, onde todas elas fazem parte de uma rede. A *Home Machine* age como um coordenador que gerencia a transferência e execução de tarefas para as *Grid Machines* e recebe os

resultados. As *Grid Machines* são as máquinas que realmente executam as tarefas e em seguida, retornam o resultado para a *Home Machine*.

2.2 Algoritmos TDIDT (*Top-Down Induction of Decision Trees*) de Geração de Árvore de Decisão

Os algoritmos TDIDT geram árvores de decisão (AD), ou seja, árvores onde os nós representam atributos, suas ramificações são valores para o atributo pai e os nós folha representam a classificação. O algoritmo SPRINT é um exemplo de algoritmo TDIDT, gerador de árvore de decisão.

Na Figura 1, podemos observar um conjunto de treinamento com os atributos idade e tipo de carro e a classe risco, e a árvore de decisão que foi gerada a partir da análise dos dados. A árvore apresenta dois pontos de particionamento: Idade < 25 e Tipo de Carro = Esporte. Podemos observar também que a condição Idade < 25 foi suficiente para determinar que o valor da classe Risco seria Alto.

SPRINT utiliza um conjunto de treinamento com exemplos de atributos/valores e a classificação do exemplo. O algoritmo é executado com o objetivo de analisar os diversos exemplos e gerar a partir desses exemplos, a menor árvore de decisão possível. A ordem em que os atributos aparecem na árvore e a condição de particionamento dos nós em cada nível da árvore depende do cálculo de funções específicas de cada algoritmo para a escolha do melhor atributo. É importante destacar que o conjunto de treinamento será particionado até que todos os atributos (nós da árvore de decisão) apresentem valores de uma mesma classificação em seus ramos.

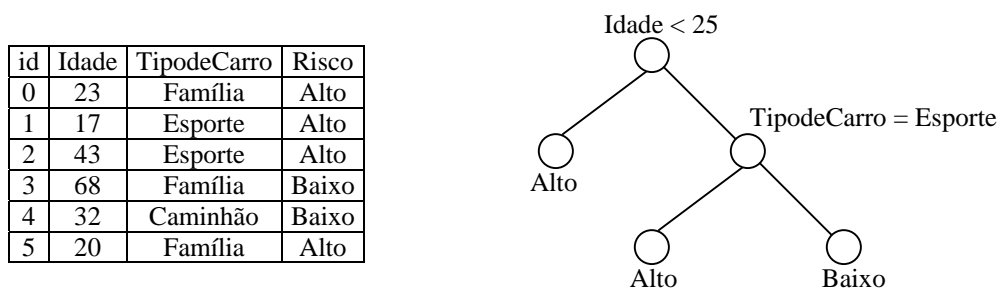


Figura 1 Conjunto de Treinamento e Árvore de decisão.

O algoritmo SPRINT utiliza estruturas de dados como listas de atributos para cada atributo do conjunto de treinamento e histogramas ou matrizes de contagem com a frequência relativa dos valores do atributo para a classe.

Na Figura 2, podemos observar a lista do atributo Tipo de Carro com o índice de cada registro, o valor do atributo (Esporte, Família ou Caminhão) e o valor da classe Risco (Alto ou Baixo). À esquerda, temos a matriz de contagem do atributo Tipo de Carro com a frequência da classe para cada valor do atributo.

Primeiramente, o conjunto de dados é analisado, as listas de atributos são separadas e seus histogramas atualizados. A partir dos histogramas, uma função é usada para a escolha do melhor atributo para particionar a árvore. No caso do algoritmo SPRINT, o cálculo utilizado é o *gini index* [3]. A função do *gini index* retorna um valor para cada atributo e o menor valor indica o melhor atributo para particionar a árvore naquele momento. A cada nova partição, os dados são novamente analisados, a função é novamente calculada para os atributos restantes e um novo atributo é selecionado para um novo particionamento. O processo é repetido até que todos os ramos levem a uma mesma classificação.

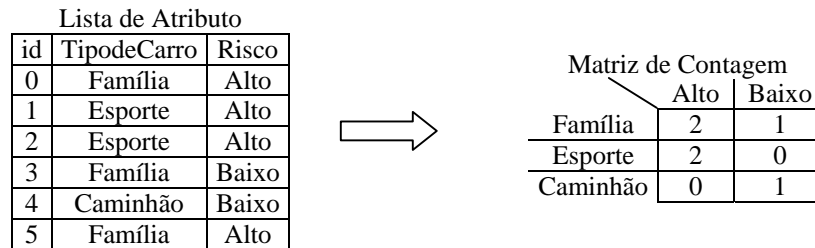


Figura 2 Estruturas de Dados

3. Paralelização de SPRINT em MyGrid

Para a paralelização do algoritmo SPRINT em MyGrid, temos N *grid machines* conectadas em uma rede e comunicando-se através de troca de mensagens. Um dos problemas da paralelização do algoritmo é garantir o balanceamento da carga de dados entre os processadores, ou seja, os dados do conjunto de treinamento usados no algoritmo devem ser distribuídos igualmente entre as N *grid machines* de MyGrid.

Como MyGrid trabalha com conjuntos de tarefas (BoT), cada passo do algoritmo deve ser executado como uma tarefa independente enviada da *home machine* para alguma *grid machine* ou uma resposta enviada de uma *grid machine* para a *home machine* informando o resultado da tarefa requisitada.

Primeiramente, o conjunto de treinamento é distribuído entre as N *grid machines*, de forma que cada *grid machine* possa trabalhar com $1/N$ do conjunto de treinamento completo. A seguir, cada *grid machine* gera suas próprias listas de atributos e histogramas correspondentes. Com as estruturas de dados completas, a *home machine* demanda então às suas *grid machines* qual o atributo escolhido como o melhor atributo para particionar a árvore. Para isso, a *home machine* envia um conjunto de tarefas (BoT) para as *grid machines*: percorrer listas de atributos, calcular o *gini index* para cada uma das listas de atributos, guardar o menor índice e enviar o resultado para a *home machine*. Com base nos resultados de todas as *grid machines*, a *home machine* deve então analisar os resultados e verificar qual o atributo é o melhor para o particionamento da árvore naquele momento. Uma vez verificado o melhor atributo, a *home machine* envia o resultado para as *grid machines* para que estas possam atualizar suas listas de atributos eliminando o atributo já utilizado para geração da árvore.

A cada nova iteração, o índice para o cálculo do melhor atributo deve ser novamente computado e enviado para a *home machine* por cada uma das *grid machines*. Somente após a resposta da *home machine* é que as *grid machines* podem seguir com a geração da árvore de decisão. Devemos observar que o conjunto de treinamento é dividido uma única vez e a

cada atributo analisado, as *grid machines* devem apenas atualizar suas listas de atributos e não gerá-las novamente. Assim, o tráfego na rede fica bastante reduzido pois o volume de dados trafegando na rede é bem pequeno. Não há necessidade de reenviar os dados do conjunto de treinamento. As únicas informações que devem ser repassadas são o *gini index* (das *grid machines* para a *home machine*) e o melhor atributo (da *home machine* para as *grid machines*).

4. Resultados e Trabalhos Futuros

Este trabalho de pesquisa busca avaliar a performance da paralelização do algoritmo SPRINT em termos do tempo de execução e a consequente acurácia da árvore de decisão gerada.

Os testes realizados nesse sentido consistiram em executar o algoritmo SPRINT tanto de forma serial como também paralelizada no MyGrid. As máquinas utilizadas nos testes foram um Pentium4 de 2 MHz com 512Mbytes de memória. Na execução paralela em MyGrid foram utilizadas 6 máquinas, sendo uma *Home Machine* e 5 *Grid Machines*, todas com essa mesma configuração. No caso da execução serial somente uma máquina foi usada. Vale ressaltar que os valores de tempo apresentados podem variar de acordo com a máquina escolhida.

Utilizando bases de dados de tamanhos diferentes, após cada execução foi aferido o tempo consumido e a acurácia do resultado. A acurácia foi obtida usando o método *K-Fold Cross Validation* [6].

Para que uma possível sobrecarga momentânea não interferisse nos resultados da avaliação, cada execução foi repetida 10 vezes. Ou seja, a análise realizada nesse trabalho considerou os valores médios do tempo de execução e da acurácia.

Nos testes aqui apresentados, foram utilizadas duas bases de dados obtidas do *UCI Machine Learning Repository* [7]. A primeira base de dados (BRIDGES) com 108 observações enquanto a segunda (NURSERY) com 12960.

De acordo com os resultados da avaliação, pôde-se comprovar, em função da acurácia, que as árvores de decisão geradas paralelamente usando MyGrid são similares àquelas geradas quando utilizando uma única máquina de forma serial. Portanto, as análises apresentadas daqui adiante consideram somente o fator tempo de execução.

Relacionando o tempo de execução com a quantidade de exemplos utilizados, os testes apontam que o algoritmo paralelizado apresenta performance inferior, em relação à sua execução serial, quanto menor for a quantidade de observações. Em outras palavras, a implementação paralela do SPRINT usando MyGrid também sugere que à medida que a quantidade de observações aumenta, o algoritmo apresenta performance quase que linear[2].

A partir da realização destes testes preliminares, pôde-se vislumbrar alguns trabalhos futuros. Um deles é a análise da escalabilidade e os fatores que interferem na velocidade de execução do algoritmo SPRINT em MyGrid. Com essa análise seria possível indicar com maior fidelidade o uso de SPRINT com MyGrid para classificar bases de dados extensas. Para tanto, outros testes com mais bases de dados diferentes são imprescindíveis, assim como diferentes configurações de processadores em MyGrid devem ser utilizados.

A implementação da paralelização de SPRINT usando MyGrid constitui um avanço nas possibilidades de tratamento de grandes volumes de dados. Em vista disso, percebe-se que outros algoritmos também poderiam ser beneficiados com o uso de MyGrid. Portanto,

outro trabalho futuro seria comparar a performance da paralelização do algoritmo SPRINT com outros algoritmos paralelos também utilizando MyGrid.

5. Conclusão

Com a necessidade cada vez maior de processar e analisar bases volumosas de dados, as aplicações de Mineração de Dados são cada vez mais utilizadas.

Nesse trabalho, apresentamos o algoritmo SPRINT, utilizado para mineração de dados, e sua paralelização na plataforma MyGrid para oferecer melhor desempenho para as aplicações.

Temos o objetivo de continuar esse trabalho de pesquisa, realizando testes para comprovar a acurácia e o melhor desempenho de SPRINT em MyGrid e comparar também a performance do algoritmo SPRINT com outros algoritmos da família TDIDT já testados em MyGrid.

6. Referências

[1] M. Mehta, R. Agrawal, and J. Rissanen. SLIQ, *a fast scalable classifier for data mining*. In Proceedings of the Fifth International Conference on Knowledge Discovery in Databases and Data Mining, Montreal, Canada, August 1995.

[2] J. C. Shafer, R. Agrawal, and M. Mehta. *SPRINT: A scalable parallel classifier for data mining*. In Proceedings of the Twenty-Second International Conference on Very Large Databases, pages 544-555, Bombay, India, 1996. Morgan Kaufmann.

[3] Breiman L, Friedman JH, Olshen RA, and Stone CJ. *Classification and regression trees*. Belmont, CA: Wadsworth International Group, 1984.

[4] OurGrid Home Page. <http://www.ourgrid.org/>

[5] Globus Project. <http://www.globus.org/ogsa/>

[6] Y. Bengio and Y. Grandvalet. *No unbiased estimator of the variance of K-fold cross-validation*. Journal of Machine Learning Research, 2003.

[7] UCI Machine Learning Repository.
<http://www.ics.uci.edu/~mllearn/MLRepository.html>