7th Brazilian Conference on Dynamics, Control and Applications

May 07 - 09, 2008
FCT - Unesp at Presidente Prudente, SP, Brazil

DINCON' 2008

MINERAÇÃO DE DADOS PARALELA EM CLUSTER DE COMPUTADORES: UM TESTE INICIAL COM ANÁLISE DE DADOS PLUVIOMÉTRICOS DINCON' 2008

Fábio A. Faria [1], *Milton H. Shimabukuro* [2], *Erwin Doescher* [2], *Edilson F. Flores* [2]

[1] Instituto de Computação-Unicamp, Campinas-SP, Brazil, fabio.faria@students.ic.unicamp.br
[2] Faculdade de Ciências e Tecnologia-Unesp, Pres. Prudente-SP, Brazil, {miltonhs,erwin,efflores}@fct.unesp.br

Abstract: This paper presents the initial tests with K-Means clustering algorithm with the aim of implementing parallel data mining on computer cluster environment. Tests were done with pluviometric data set and other types of data specifically generated. Comparision with results from sequential execution were realized too.

Keywords: Parallel Data Mining, Computer Cluster, K-Means Algorithm

1. INTRODUÇÃO

A Mineração de Dados proporciona detectar o "esperado" e descobrir o "inesperado", isto é, extrair informação, a partir de um conjunto de dados volumoso, com o objetivo de chegar a conhecimento anteriormente não previsto. Nos dias atuais, no âmbito comercial, as empresas estão dando um valor ainda maior para suas bases de dados, pois sabem que no mercado se destacam e se sobressaem aquelas que conseguem obter o maior conhecimento sobre seus clientes.

Pesquisas em Mineração de Dados surgiram pela necessidade de adaptação ou criação de técnicas e ferramentas para análise de grande volume de dados, viabilizado pela capacidade crescente de geração e armazenamento de dados que aumenta dia a dia. Tal volume, por si só, já é um obstáculo para a análise, ao qual se soma a exigência de grande capacidade de processamento. A área de Mineração de Dados é multi e interdisciplinar. Técnicas da área da Ciência da Computação podem contribuir para a análise efetiva desses dados.

Uma das formas de aumentar a velocidade é o processamento paralelo de algoritmos para análise de dados. A paralelização tem como objetivo, no contexto de análise de dados, buscar agilidade no processo de mineração propriamente dito, pois este exige um alto poder de processamento e memória. Tornar paralelo esta atividade permite processar um volume de dados ainda maior e, assim, obter maior precisão no processo de mineração.

Neste artigo são apreesentados os resultados de testes iniciais na aplicação do algoritmo de agrupamento K-Médias em um conjunto de dados pluviométricos. O algoritmo foi implementado, utilizando a biblioteca MPI, com paralelismo de dados, e testado em um ambiente de cluster de computadores. Esses dados já foram processados com software estatístico de forma convencional, isto é, execução sequencial, cujos resultados foram utilizados para comparação com aqueles de execução paralela. Busca-se, com estes resultados, direcionar outros testes, mais aprofundados, para análise de conjunto de dados mais volumosos e, para medir o impacto no tempo de processamento.

O processo de mineração de dados é abordado na Seção 2, seguido pela descrição de ambientes de execução paralela de programas de computador. A implementação do algoritmo K-Médias para ambiente de cluster de computadores e o teste com dados pluviométricos são apresentados na Seção 4, seguido pelas conclusões.

2. PROCESSO DE MINERAÇÃO DE DADOS

Na definição de Fayyad, Piatetsky-Shapiro e Smyth [3], " Extração de Conhecimento de Base de Dados (KDD, Knowledge Discovery in Databases) é o processo de identificação de padrões válidos, novos, potencialmente úteis e compreensíveis embutidos nos dados". O processo de Extração de Conhecimento de Bases de Dados tem o objetivo de encontrar conhecimento a partir de um volumoso conjunto de dados para ser utilizado em um processo decisório. Portanto, um requisito importante é que esse conhecimento descoberto seja compreensível, além de útil e interessante para os usuários finais do processo, que geralmente são tomadores de decisão, de forma que ele forneça um suporte a esses usuários no processo decisório.

Na definição acima, a Mineração de Dados (DM, Data Mining) é parte do processo de KDD; entretanto, neste texto, será adotado KDD e DM como sinônimos. A Mineração de Dados é multidisciplinar, bem com interdisciplinar, e técnicas de várias áreas podem ser aplicadas, tais como Computação de Alto Desempenho, Estatística, Bancos de Dados, Aprendizado de Máquina e Visualização.

O processo de Mineração de Dados é composto por cinco fases ilustradas na figura abaixo.

Figura 1. Processo de Mineração de Dados. Fonte: [5]

Na fase de Conhecimento do Domínio, são traçados os objetivos e as metas a serem alcançados; no Pré-Processamento ocorrem o tratamento, a limpeza e a redução do volume de dados, pois eles podem não estar adequados para o inicio do processo de mineração; na Extração de Padrões, é feita a escolha da tarefa de mineração e do algoritmo a ser utilizado; o Pós-Processamento, é naquela em que será avaliado o conhecimento extraído e por fim, a Utilização do Conhecimento. Todo o processo busca um suporte melhor fundamentado para uma tomada de decisão. O processo é interativo, ou seja, as fases podem ser refeitas tantas vezes quanto necessárias para que os objetivos sejam alcançados, chegando-se a conhecimentos válidos.

Existem alguns algoritmos paralelos para Mineração de Dados. Entre eles estão o ADDFIT (Additivi Model), no qual os custos computacionais são mais baixos, mas as aproximações são mais grosseiras, apropriadas para problemas várias dimensões; o HISURF (High Dimensional Surface Smoothing), baseado em wavelet para resolver problemas multi dimensionais; e 0 TPSFEM (Thin Plate Splines Finite Element Method), que permite obter uma aproximação mais exata com um custo mais elevado de computação, suporta problemas com duas ou três dimensões. Todos estes algoritmos seguem duas etapas, a primeira é ler os dados e montar um sistema linear e a segunda é adicionar "confinamentos" e resolver o sistema linear. A primeira etapa é fácil e eficientemente paralelizável. Os dados são lidos do disco apenas uma vez. O tamanho do sistema linear simétrico é independente do número de registros de dados. O ADDIFIT e o HISURF resultam em uma matriz densa e o TPSFEM em uma grande matriz esparsa. Todos esses algoritmos são preditivos [6].

O algoritmo escolhido para a tarefa de agrupamento de um conjunto de dados pluviométrico é o K-Médias, por ser de uso tradicional para a tarefa e permitir a comparação do resultados, por ter sido aplicado ao referido conjunto de dados.

Optou-se pelo cluster Beowulf, descrito na Seção a seguir, para a execução do algoritmo por ele ter sido criado para uso com desktop comuns, buscando uma boa relação preço/desempenho.

3. AMBIENTE PARA EXECUÇÃO PARALELA

Segundo Pitanga [4], a utilização de dois ou mais computadores em conjunto para resolver um problema é chamado de cluster. A idéia principal de um cluster é um conjunto de computadores que juntam forças para resolver um problema computacional em comum, sendo que este problema pode necessitar de um maior poder de computação ou exigir disponibilidade para o processamento de um serviço qualquer. Portanto, uma configuração de computadores em cluster pode ser utilizada com os objetivos de alto desempenho, alta disponibilidade e balanceamento de carga.

O cluster Beowulf é uma classe de cluster de computadores que é utilizado quando procura-se melhorar a relação preço/desempenho, que vem caracterizar os clusters construídos a partir da junção de máquinas desktop, ou seja, computadores pessoais, utilizando sistema operacional Linux e são dispostos por um nó mestre (front-end), como nome já diz, é o computador principal e os nós escravos (back-end), que executam tarefas enviadas pelo mestre. O projeto foi criado por Donald Becker da NASA, na época trabalhava no CESDIS (Centro de Excelência em Ciência da Informação e em Dados Espaciais), e tinha como objetivo construir uma organização paralela de baixo custo, composto por equipamentos disponíveis a qualquer usuário e que oferecem alto poder computacional. Este cluster era composto por computadores Intel DX4, conectados através de uma rede Ethernet e executando o sistema operacional Linux. A Figura 2 ilustra este tipo de cluster.

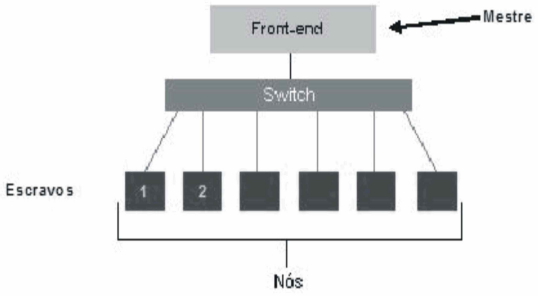

Figura 2. Equema do Cluster Beowulf

Neste projeto foi configurado um cluster Beowulf batizado de CBMDP (Cluster Beowulf para Mineração de Dados Paralela), que utiliza a distribuição Linux Debian Etch Testing, com três nós ou nodos (Mestre, Escravo1 e Escravo2), Celeron D, 512MB de memória RAM e disco rígido de 80 GB, com o objetivo de aproveitar configurações comuns de microcomputadores.

Foi utilizada a biblioteca MPI (Message Passing Interface), que tem opções como envio de mensagens broadcast (para todas as máquinas do cluster) e multicast (para um grupo específico de máquinas), assim como um melhor controle sobre o tratamento que cada mensagem terá ao ser recebida por outro ponto do cluster. A configuração do MPI depende da implementação utilizada e algumas delas chegam a instalar front-ends para compiladores em C e

Fortran, mas a forma geral de uso é semelhante [1]. Quando se programa em paralelo com a MPI, a responsabilidade de paralelização é toda do programador, os métodos disponibiizados apenas fazem com que os processos se comuniquem entre eles através de trocas de mensagens. Seu funcionamento é dividir os problemas em pequenas partes e essas partes são distribuídas para que outras máquinas do *cluster* façam o cálculo em cada uma dessas partes e, em seguida, os resultados calculados nas máquinas são enviados a uma receptora que coleta os mesmos, agrupa-os e gera o resultado final.

Alguns testes iniciais de desempenho foram realizados para uma medição de desempenho do CBMDP, instalando em seu sistema o aplicativo de renderização de imagem PovRay (Persistence of Vision Ray Trace). Foram escolhidas duas imagens para serem renderizadas, com diferentes níveis de detalhes. O que pôde ser observado foi a seguinte relação de ganho de tempo: utilizando dois processos e apenas o nó Mestre para renderizar uma das imagens, foram gastos cinco minutos e quarenta e três segundos, e com os mesmos dois processos, porém, utilizando o nó Mestre e o nós, Escravo1 e Escravo2, foram gastos quatro minutos e dezoito segundos, obtendo um ganho em torno de 25%. Com quatro processos, foram obtidos ganhos de 75% com uma das imagens, e 68% com a outra. Os resultados dos testes com o CBMDP mostram também que, o aumento somente no número de processos não implica em um ganho de desempenho, exigindo um planejamento da execução paralela , que é influenciada pela configuração do ambiente.

Na criação de programas paralelos, pode-se optar por duas formas de paralelismo, de acordo com a adequação à aplicação:

- Paralelismo de tarefa. É usado quando as partes do algoritmo que se quer paralelizar não são dependentes entre si, possibilitando a distribuição de suas partes entre os nós do cluster ou entre os processadores de um mesmo computador;
- Paralelismo de dados. É muito usado quando se tem um conjunto muito grande de dados e estes são incapazes de ser carregados todos juntos na memória do computador. Então é feita a divisão deste conjunto em subconjuntos menores para que se possa carregar-los em pedaços até que todo o conjunto de dados seja utilizado.

Nestes testes iniciais com dados pluviométricos foi empregado o paralelismo de dados. A paralelização do algoritmo K-Médias é descrita na próxima Seção.

4. USO DO ALGORITMO K-MÉDIAS NO AMBIENTE CBMDP

O método K-Médias é um algoritmo para partição de um conjunto de elementos em determinado número de agrupamentos K, utilizando o método dos mínimos quadrados e o ajustamento das médias no decorrer dos processos de iterações. Este algoritmo, de forma geral, possui obrigatoriamente apenas duas etapas. Uma delas é a determinação dos centróides, que é a média de todas as amostras de um agrupamento, sendo o número de grupos (k) definido anteriormente pelo usuário, juntamente com as "sementes", estas sendo utilizadas para a formação de novos agrupamentos. E a outra é a associação de cada elemento a uma semente mais próxima.

Existem mais de 20 tipos de algoritmos descritos como algoritmos de partições K-Médias, e o usado neste trabalho foi o de Bussab, Miazaki & Andrade [2]. Este algoritmo é conhecido por ser muito rápido e também por possuir boa propriedade de convergência.

O algoritmo escolhido foi implementado na linguagem de programação C, utilizando a biblioteca MPI, para obter o seu paralelismo. Este algoritmo é utilizado seqüencialmente, ou seja, suas partes são dependentes entre si e por isso, a escolha do paralelismo de dados. Os passos são explicados a seguir.

- Passo 1: O nó Mestre faz a leitura do conjunto de dados e o divide em duas partes, figura 3;
- Passo 2: Cada Escravo recebe uma parte dos dados enviados pelo Mestre, primeiro o Escravo1 e depois o Escravo2, além das duas sementes escolhidas, que servirão de base para cada um dos grupos, figura 3;
- Passo 3: Nos Escravos, aplica-se o algoritmo K-Médias sobre o conjunto de dados e são formados dois agrupamentos com estes dados. Os dois grupos formados são enviados para o Mestre em ordem de recebimento, primeiro o Escravo1 e depois o Escravo2. Lembrando que o algoritmo é executado simultaneamente nos dois nós Escravos, figura 4;
- Passo 4: No nó Mestre, os grupos do Escravo1 (Y1 e Y2) também formarão os grupos do Mestre (Grupo 1 e Grupo 2) e o posicionamento dos elementos dos grupos Z1 e Z2 é realizado em função das sementes definidas anteriormente. Por exemplo, se uma semente que estiver no grupo Y1 e também estiver no grupo Z1, todos os elementos desses dois grupos se agruparão formando o Grupo 1 do Mestre e se a semente do grupo Y2 também constar no grupo Z2, esses dois grupos se agruparão para formar o Grupo 2 do Mestre.

Logo, no Mestre o Grupo 1 será formado pelos grupos Y1 e Z1 e o Grupo 2 será formado pelos grupos Y2 e Z2. Existem quatro combinações para o resultado final, que estão apresentadas na figura 4.

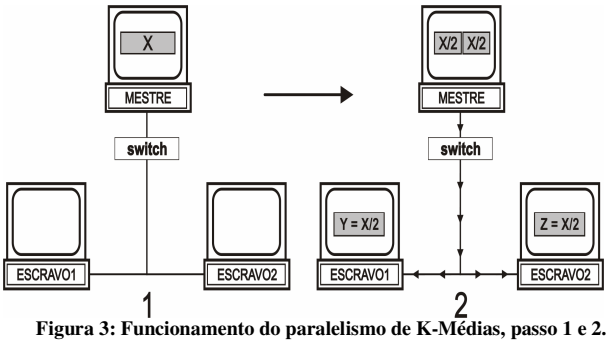

Figura 3: Funcionamento do paralelismo de K-Médias, passo 1 e 2.

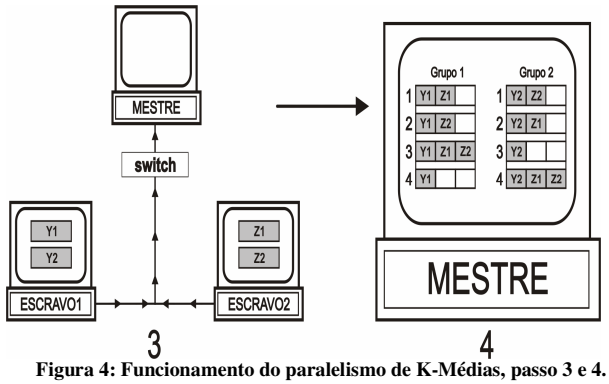

Figura 4: Funcionamento do paralelismo de K-Médias, passo 3 e 4.

Foi realizado um teste com uma grande quantidade de dados, um milhão (1.000.000) de registros. Utilizando o software estatístico MINITAB, foi gerado randomicamente os dados, realizou-se a tarefa de agrupamento com duas variáveis (Altura e Peso) em dois grupos. Com este conjunto de dados, foi utilizado o algoritmo paralelo implementado e os resultados foram exatamente iguais sem o uso do algoritmo, porém o tempo de processamento no algoritmo paralelo foi menor, logo o funcionamento do algoritmo implementado pode ser considerado correto e mais eficaz para este exemplo, embora esta afirmação não possa ser generalizada, ou seja, afirmar que o algoritmo funcionará para qualquer tipo de problema.

4.1 TESTES COM DADOS CLIMATOLÓGICOS

Foram realizados testes do algoritmo K-Médias implementado com dados pluviométricos do Estado de São Paulo disponibilizados pelo Departamento de Água e Energia Elétrica (DAEE). O teste possibilitou a formação de dois grupos com 998 estações pluviométricas distribuídas em todo o Estado, ver figura 5.

Na Tabela 1, são mostradas seis colunas do conjunto de dados (Estações Pluviométricas, Verão, Outono, Inverno, Primavera e Total Anual). As estações pluviométricas são estruturas físicas de coleta de dados que se encontram em diversas regiões do Estado de São Paulo, a coluna Verão é calculada pela soma dos 3 primeiros meses do ano de chuva, o Outono é a soma do quarto ao sexto mês de chuva, o Inverno é a soma do sétimo ao nono mês de chuva e a Primavera é a soma dos três últimos meses do ano, e a coluna Total Anual é a soma do total das quatro estações do ano.

Tabela1: Formato do conjunto de dados do Estado de São Paulo.

Estações Pluviomé-tricas	Verão	Outono	Inverno	Prima-vera	Total Anual
A6-001	462.1	7.6	45.5	451.7	966.9
B4-028	590.8	49.1	61.7	615.4	1317.0
B4-029	596.3	33.7	69.2	601.9	1301.1
…	…	…	…	…	…

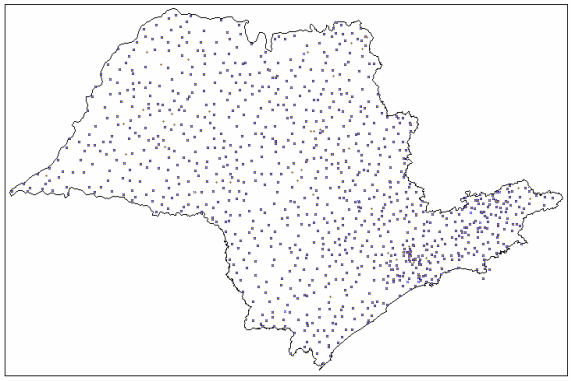

Figura 5. Estações Pluviométricas do Estado de São Paulo no ano 2003.

No teste foi tomado como critério para agrupamento uma das estações, considerando as colunas outono, inverno, verão e primavera. As outras estações são consideradas variáveis. No teste inicial também foi considerado o total anual.

Uma parte dos resultados é apresentada nas Tabelas 2 e 3.

Tabela 2: Elementos do Grupo 1.

	Verão	Outono	Inverno	Primavera	Total Anual
1	A6-001	A6-001	A6-001	A6-001	A6-001
2	B7-013	B4-027	B4-027	B7-012	B7-013
3	B7-014	B4-038	B7-014	B4-032	B7-014
4	B4-038	B4-032	B7-012	B5-021	B4-038

Tabela 3: Elementos do Grupo 2.

	Verão	Outono	Inverno	Primavera	Total Anual
1	B4-028	B4-028	B4-028	B4-028	B4-028
2	B4-029	B4-029	B4-029	B4-029	B4-029
3	B4-026	B4-026	B4-026	B4-026	B4-026
4	B4-027	B7-013	B7-013	B7-013	B4-027

Analisando esses resultados, verifica-se que todos os grupos são diferentes, este comportamento se deve aos diferentes critérios de agrupamentos, o agrupamento das estações pluviométricas também ficou diferente. Outro fato observado foi que no grupo 1 as variáveis Verão e Primavera são bem semelhantes a variável Total Anual, a justificativa é que no verão e na primavera os níveis pluviométricos são bem elevados e com isso, as variáveis tendem a influenciarem no valor Total Anual. Além disso, nota-se que as estações pluviométricas que pertencem ao mesmo grupo são também geograficamente próximas uma das outras, como é o caso das estações B4-028 e B4-029, no grupo 2 da variável Verão.

5. CONCLUSÃO

O processo de Mineração de Dados é interativo e iterativo, e requer um alto poder de processamento e grande disponibilidade de memória para que se possa analisar grandes quantidades de dados. Grandes volumes de dados são comuns à várias aplicações, considerando a capacidade crescente de geração, captação e armazenamento de dados, o que deixa um amplo escopo de aplicação para mineração de dados.

O estudo da paralelização no processo de Mineração de Dados é importante para criar soluções alternativas na busca de prover maior agilidade ao processo. Pelos resultados iniciais, foi possível considerar a paralelização como uma alternativa viável (relação custo/benefício, em virtude da configuração usual dos equipamentos) e adequada para a análise de grandes volumes de dados, que engloba tarefas que necessitam de alto poder de processamento e memória. A utilização da biblioteca MPI facilitou muito a implementação do algoritmo paralelo, devido ao seu suporte a linguagem de programação C, de amplo uso na área.

O algoritmo K-Médias, escolhido por sua tradicional aplicação à tarefa de agrupamento e familiaridade aos usuários, foi adaptado como solução paralelizada para um problema específico, e comportou-se da forma esperada, pois gerou o mesmo resultado do processamento seqüêncial. Entretanto, uma generalização para outros problemas não é imedidata, o que reqüer um estudo não só da paralelização, mas também da forma de coordenação dos trechos executados em paralelo, bem como uma medição mais acurada do ganho de desempenho.

REFERÊNCIAS

[1] BUENO, A.D. (2002). Introdução ao Processamento Paralelo e ao Uso de Clusters de Workstations em Sistemas GNU/LINUX - Filosofia, Laboratório de Meios Porosos e Propriedades Termofísicas, Universidade Federal de Santa Catarina.

[2] BUSSAB, W. O., MIAZAKI, E.S. & ANDRADE, D. F. (1990). Introdução à Análise de Agrupamentos 58-63.

[3] FAYYAD, U., PIATETSKY-SHAPIRO, G. e SMYTH, P. (1996). From Data Mining to Knowledge Discovery: An Overview. In: Advances in Knowledge Discovery & Data Mining, pp.1–34.

[4] PITANGA, M. et. al. (2004). Construindo Supercomputadores com Linux, Editora Brasport Manole Ltda.

[5] REZENDE, S. O. *et al.* (2003). Mineração de Dados. In Rezende, S. O. (org.) Sistemas Inteligentes: Fundamentos e Aplicações, Capítulo 12, pp. 307-33, Editora Manole Ltda.

[6] STRAZDINS, P., CHRISTEN, P., NIELSEN, O. M. E HEGLAND, M. (2001). Parallel Data Mining on a Beowulf Cluster, 5th Int'l Conference on High-Performance Computing in the Asia-Pacific Region, Queensland, Australia.