ParGRES: uma camada de processamento paralelo de consultas sobre o PostgreSQL

Marta Mattoso[1], Geraldo Zimbrão[1,3], Alexandre A. B. Lima[1], Fernanda Baião[1,2], Vanessa P. Braganholo[1], Albino A. Aveleda[1], Bernardo Miranda[1], Bruno Kinder Almentero[1], Marcelo Nunes Costa[1]

[1]Programa de Sistemas e Computação e NACAD, COPPE – UFRJ
[2]Departamento de Informática Aplicada – UNIRIO [3]Departamento de Ciência da Computação/IM – UFRJ

pargres@yahoogroups.com, http://pargres.nacad.ufrj.br

Abstract. *ParGRES aims at the development of free software to efficiently process heavy weight queries involving large databases by using PostgreSQL DBMS on top of PC clusters. ParGRES is a middleware based on intra- and inter parallel query processing and uses database replication combined with virtual fragmentation.*

Resumo. *ParGRES é um projeto que tem como objetivo desenvolver um sistema de software livre para processar com eficiência consultas pesadas que envolvam grandes quantidades de dados, usando para isso, o SGBD PostgreSQL sobre clusters de PCs. No ParGRES, o processamento da consulta explora o paralelismo intra- e inter-consultas, usando replicação e fragmentação virtual de dados.*

1. Introdução

Com o crescente volume de dados manipulado por sistemas de Tecnologia da Informação, diversas empresas têm tido dificuldade em atender ao correspondente aumento nos custos de hardware e de software de sistemas de banco de dados.

Aplicações OLAP (*On-Line Analytical Processing*) necessitam de suporte de alto desempenho para operações em bancos de dados. Tipicamente, essas aplicações acessam grandes conjuntos de dados através de consultas de alto custo e de natureza *ad-hoc*. Operações de atualização são menos freqüentes. O *benchmark* TPC-H [7],voltado para representar genericamente aplicações OLAP, apresenta vinte e quatro operações de acesso ao banco de dados. Entre elas, vinte e duas são consultas de alto custo enquanto apenas duas são atualizações. A otimização de bancos de dados para suporte a consultas *ad-hoc* é mais complexa, uma vez que elas não são predefinidas [3]. Torna-se então mais difícil a escolha das estruturas de acesso a serem criadas e das formas de organização dos registros em disco entre outras.

O processamento paralelo é um candidato natural para aumentar o desempenho de consultas OLAP. Entretanto, devido ao alto custo de ambientes computacionais com grande poder de processamento, uma alternativa mais barata é a utilização de *clusters* de PCs. Porém, os custos de softwares para processar grandes massas de dados são muito altos, mesmo para *clusters*.

Alguns softwares de fabricantes de Sistemas Gerenciadores de Bancos de Dados (SGBD) já oferecem recursos que permitem a exploração do processamento paralelo em operações de banco de dados sobre clusters. Entretanto, esses produtos têm custos extremamente elevados, seja pelo próprio software ou pelo hardware específico requisitado, como unidades centrais de armazenamento (SAN – *Storage Area Network*).

O ParGRES se enquadra na proposta dos chamados "clusters de bancos de dados" [1] que é a de utilizar SGBDs seqüenciais nos nós de um cluster como componentes do tipo "caixa-preta", ou seja, sem modificar seu código-fonte para incluir funcionalidades que melhorem sua utilização em clusters. Essa abordagem evita as dispendiosas operações de migração de bancos de dados, normalmente necessárias nos SGBDs paralelos. Um cluster de BD consiste numa camada de software intermediária (*middleware*), entre a aplicação e os SGBDs, capaz de coordenar as operações dos mesmos a fim de obter o paralelismo desejado utilizando um cluster de PCs padrão.

O paralelismo do ParGRES é voltado para as consultas pesadas típicas de aplicações OLAP e propomos uma solução para essa demanda implementando paralelismo intra-consulta em um cluster de BD. O paralelismo intra-consulta significa decompor consultas complexas em sub-consultas que serão executadas em paralelo. A idéia é que cada sub-consulta atue em um fragmento de dados diferente. Dessa forma, cada sub-consulta poderá então ser enviada para o nó que possui o respectivo fragmento dos dados. Assim, cada sub-consulta é enviada para um nó diferente e executada em paralelo com as demais. Embora estejamos realizando o desenvolvimento sobre o PostgreSQL [6], o paralelismo é baseado no padrão SQL, não sendo dependente de nenhuma característica específica do SGBD.

As principais soluções de clusters de BD são PowerDB [1] e C-JDBC [2]. Entretanto, apenas o PowerDB oferece o paralelismo intra-consulta, mas trata-se de software proprietário. O C-JDBC é software livre, porém se baseia no paralelismo inter-consultas. Esse tipo de paralelismo possibilita a obtenção de alto desempenho no processamento de números elevados de pequenas transações concorrentes, típicas do ambiente OLTP (*On-Line Transaction Processing*). Porém, são consultas de alto custo, típicas do ambiente OLAP, que costumam apresentar longo tempo de processamento mesmo se executadas isoladamente. Nesse caso, a solução inter-consultas não é atraente, pois não consegue reduzir esse tempo.

O ParGRES conta com financiamento da Finep e Itautec através do Edital de Software Livre CT-INFO - 01/2003 e está sendo desenvolvido em Java com base no SmaQ, um protótipo desenvolvido previamente [5]. Pretende-se que em dezembro o software seja publicado num portal de software livre.

Este trabalho está organizado conforme a seguir. A seção 2 descreve as características da arquitetura e a seção 3 descreve as perspectivas e como pretendemos avaliar a eficiência do ParGRES. Finalmente a seção 4 conclui.

2. A Arquitetura do ParGRES

Como as demais soluções para clusters de bancos de dados, o ParGRES consiste em uma camada intermediária de software (*middleware*) que orquestra instâncias de SGBDs em execução nos diferentes nós do cluster para a implementação de técnicas de

processamento paralelo. Em vários projetos, como o C-JDBC [2] e o PowerDB [1], essa camada é centralizada e executada em um nó escolhido como coordenador. O ParGRES, no entanto, possui arquitetura descentralizada, como ilustra a Figura 1. Seus diferentes componentes se encontram distribuídos entre os nós do cluster. A arquitetura da Figura 1 foi proposta em [4] e parcialmente implementada no protótipo SmaQ, cujo foco foi a avaliação do paralelismo intra- e inter-consultas. Enquanto que em SmaQ não há suporte a atualizações, o módulo Intermediador não está implementado e o CQP não re-escreve a paralelização das consultas automaticamente, pretende-se com ParGRES realizar a implementação do sistema como um todo incluindo os módulos específicos para controle de concorrência junto ao suporte a operações de atualização.

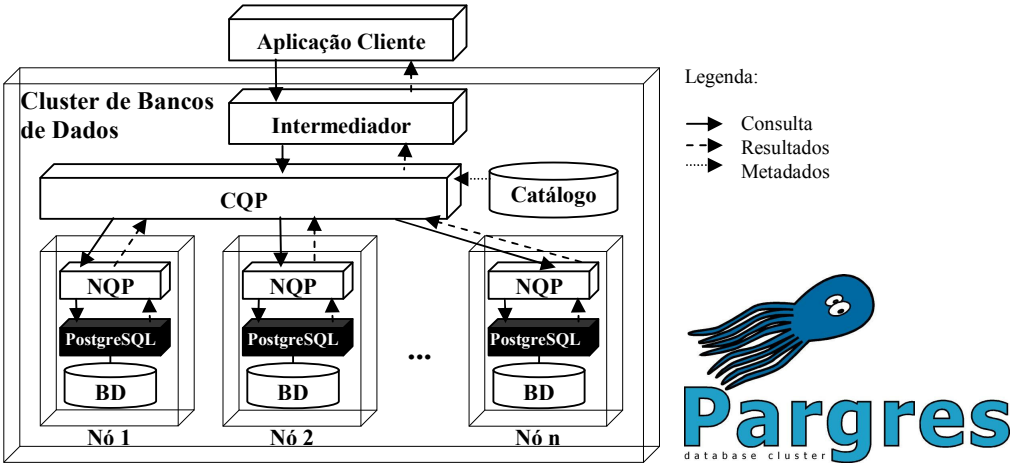

Figura 1 - Visão Geral da Arquitetura do ParGRES

Há basicamente componentes de dois tipos: *globais* e *locais*. Componentes globais são aqueles que executam tarefas que envolvem vários nós do cluster, enquanto os locais executam tarefas em apenas um nó. Os componentes globais são o Intermediador e o Processador de Consultas de Cluster (CQP – *Cluster Query Processor*). Os componentes locais são o Processador de Consultas de Nó (NQP – *Node Query Processor*) e o SGBD PostgreSQL.

O componente mais importante e que atua como coordenador de todos os demais é o CQP. Ele é o responsável pelo controle da execução de requisições no ParGRES. Como a maioria dos clusters de PCs possuem um único nó (nó de entrada) acessível por aplicações externas, o CQP deveria ser sempre alocado nesse nó, uma vez que deve se comunicar com as aplicações. Para prover maior flexibilidade na alocação do CQP, utilizamos o componente Intermediador, cuja função é repassar as requisições das aplicações para o CQP e, no sentido inverso, repassar as respostas do CQP às aplicações. Com isso, temos total flexibilidade na alocação física do CQP, o que aumenta o grau de tolerância a falhas do ParGRES. Além disso, prepara a arquitetura para uma futura implementação distribuída do CQP.

Há três tipos de tarefas executadas pelo ParGRES: processamento de consultas com paralelismo inter-consultas, processamento de consultas com paralelismo intra-

consulta e processamento de operações de atualização de dados. O objetivo principal é fornecer alto desempenho para o processamento de consultas de alto custo. Para tanto, o ParGRES implementa o paralelismo intra-consulta utilizando a técnica de fragmentação virtual adaptativa [4] com replicação total de dados. Como várias consultas podem ser processadas simultaneamente no cluster, inclusive pelos mesmos nós, o paralelismo inter-consultas é empregado juntamente com o intra-consulta. Uma das vantagens dessa combinação é que algumas consultas podem ser de baixo custo, não justificando a utilização de paralelismo intra-consulta, bastando a utilização de paralelismo inter-consultas. O CQP é o responsável por analisar as consultas e decidir que tipo de paralelismo será utilizado no processamento de cada uma bem como os nós utilizados nesse processamento. Para tanto, utiliza informações presentes no seu Catálogo. Esse catálogo é, no entanto, muito simples e armazena apenas as informações necessárias à implementação da fragmentação virtual adaptativa, que é uma técnica não intrusiva. Sendo assim, esse catálogo não necessita de informações específicas do SGBD PostgreSQL, mantendo a filosofia de se utilizar o SGBD como um componente do tipo "caixa-preta".

Apesar do foco principal do ParGRES ser o processamento de consultas, operações de atualização de dados podem ser enviadas pela aplicação cliente. A execução paralela de operações de atualização e de consultas com paralelismo intra-consulta em um ambiente com replicação total de dados poderia gerar resultados inconsistentes. Como atualizações de dados em um ambiente OLAP são realizadas, tipicamente, em momentos predeterminados [3], o ParGRES adota uma política conservadora e não permite a execução paralela de atualizações e consultas. Além disso, enquanto que as atualizações são rápidas, as consultas possuem um tempo de processamento significativo [7]. Para tanto, o CQP possui um *escalonador*, responsável por ordenar consultas e atualizações. Enquanto atualizações são processadas, todas as demais operações no cluster são bloqueadas. Quando só há consultas, o CQP permite suas execuções em paralelo.

Após determinar o tipo de paralelismo e os nós envolvidos no processamento de uma requisição, o CQP inicia seu processamento. No caso de uma consulta sem paralelismo intra-consulta, o CQP apenas a envia ao NQP do nó escolhido para a execução. O NQP repassa a consulta ao PostgreSQL, que a processa. O resultado segue então o caminho inverso até a aplicação cliente. No caso de uma operação de atualização, o CQP bloqueia a execução de consultas, e a envia ao NQP de cada nó. Após a confirmação do processamento da operação por todos os NQPs, o CQP libera novamente a execução de consultas.

No caso de uma consulta a ser processada com paralelismo intra-consulta, o CQP atua em conjunto com os NQPs para a produção do resultado final. Inicialmente, ele aloca os NQPs e envia a cada um deles um plano de execução local para a consulta. Cada NQP processa localmente o seu plano e gera um resultado parcial, que é enviado ao CQP para a composição do resultado final da consulta. Após receber os resultados parciais de todos os NQPs, o CQP finaliza a composição de resultados e os envia à aplicação cliente.

Os NQPs assumem outro papel muito importante na implementação do paralelismo intra-consulta: são os responsáveis pela implementação de uma técnica de

balanceamento dinâmico que tem por objetivo equilibrar a carga dos nós envolvidos no processamento de uma mesma consulta. Devido a características dos dados utilizados por uma consulta, as cargas de trabalho inicialmente recebidas pelos nós podem ser bastante desiguais, ocasionando má utilização dos recursos de processamento do cluster. O fato de utilizarmos técnicas não intrusivas que mantêm a filosofia de utilização do SGBD como um componente "caixa-preta" dificulta a obtenção de uma divisão inicial de trabalho equânime. O objetivo do balanceamento dinâmico é corrigir essa distorção. São os NQPs, trocando mensagens entre si, que promovem esse balanceamento implementando uma técnica distribuída proposta em [5]. Devido a limitações de espaço, não a descrevemos aqui. Os resultados apresentados no referido trabalho mostram se tratar de uma técnica bastante eficiente, especialmente em situações de extrema desigualdade de cargas iniciais.

3. Validação do sistema proposto

O ParGRES se encontra em fase inicial de desenvolvimento. Não possuímos ainda dados relativos a experimentos realizados com ele. Nessa seção, descrevemos os tipos de validação que pretendemos realizar bem como resultados preliminares descritos em [5], trabalho que propõe as técnicas de processamento de consultas utilizadas no ParGRES.

Como o ParGRES se destina basicamente a aplicações OLAP, utilizaremos o *benchmark* TPC-H [7] como aplicação alvo. Trata-se de um *benchmark* de suporte à decisão definido pelo Conselho de Desempenho de Processamento de Transações (Transaction Processing Performance Council – TPC). Ele é constituído por um conjunto de consultas *ad-hoc* orientadas a negócios, com operações de alteração de dados concorrentes, voltado para sistemas de suporte à decisão.

O esquema da base de dados do TPC-H é composto por oito tabelas, divididas entre duas tabelas de fatos - Orders e Lineitem - e outras seis de dimensão - Region, Nation, Supplier, Part, Customer e Partsupp. Com exceção das tabelas Region e Nation, toda a cardinalidade das outras tabelas é determinada pelo fator de escala definido na geração da base de dados. Este fator é usado para separar os resultados do benchmark em classes comparativas de acordo com o espaço em disco que ocupa, desde 1 Gb até 10.000 Gb. A Figura 2 ilustra o esquema com suas tabelas e suas devidas cardinalidades parametrizadas pelo fator.

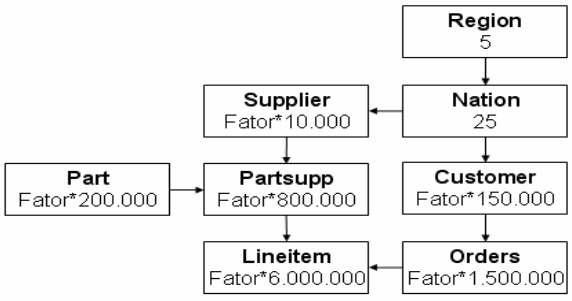

Figura 2 - Esquema do TPC-H

Pretendemos avaliar o desempenho do ParGRES em vários aspectos. Primeiramente, avaliaremos a aceleração conseguida pelo emprego do paralelismo intra-

consulta no processamento de consultas isoladas. Várias técnicas a serem utilizadas no ParGRES, como a fragmentação virtual adaptativa e a redistribuição dinâmica de carga, já foram implementadas no SmaQ [5]. Os resultados obtidos pelo SmaQ foram excelentes, alcançando aceleração linear e superlinear em vários casos no processamento de consultas [4],[5], onde experimentos com até 64 nós e base TPC-H com fator de escala 5 são retratados. Esperamos obter resultados semelhantes com o ParGRES utilizando o cluster da Itautec.

Outro aspecto a ser avaliado é a vazão obtida durante o processamento simultâneo de diferentes consultas. Tal aspecto também foi investigado no SmaQ e os resultados obtidos mostraram que o paralelismo intra-consulta proporciona vazão muito maior do que a obtida apenas com paralelismo inter-consulta em cenários OLAP típicos [5]. Mais uma vez, esperamos obter resultados semelhantes já que utilizamos as mesmas técnicas básicas em ambos os trabalhos.

O último aspecto que pretendemos avaliar é a vazão do sistema durante a execução simultânea de operações de consulta e atualização de dados. Esse aspecto não foi avaliado no SmaQ, uma vez que SmaQ não possui suporte a atualizações.

4. Conclusão

O ParGRES possui relevância estratégica por visar atender uma demanda crescente, em diversos setores de aplicação, por SGBD de alto desempenho sobre grandes massas de dados. O aspecto inovador desta proposta reside na integração de tecnologias de clusters de BD para processamentos de consultas pesadas como em OLAP em um componente de baixo custo, tecnologia aberta, que seja complementar ao SGBD seqüencial livre PostgreSQL ao mesmo tempo em que oferece uma redução significativa no tempo de execução dessas consultas.

Referências

[1] Akal, F., Böhm, K., and Schek, H.-J. (2002), "OLAP Query Evaluation in a Database Cluster: a Performance Study on Intra-Query Parallelism", In: Proceedings of the East European Conf. on Advances in Databases and Information Systems (ADBIS), 6th European East Conference, Bratislava, Slovakia, Sep, pp. 218-231.

[2] Cecchet, E., Marguerite, J., and Zwaenepoel, W. (2004), "C-JDBC: Flexible Database Clustering Middleware", In: Freenix 2004: USENIX Annual Technical Conference, Boston, MA, USA, June, pp. 9-18.

[3] N. Gorla, "Features to Consider in a Data Warehousing System", Communications of the ACM, 46(11), November, 2003, pp. 111-115.

[4] Lima, A. A. B., Mattoso, M. and Valduriez, P. (2004), "Adaptive Virtual Partitioning for OLAP Query Processing in a Database Cluster", In: Proceedings of the 19th Brazilian Symposium on Databases, Brasília, Brazil, Oct, pp. 92-105.

[5] Lima, A. A. B. (2004), "Paralelismo Intra-Consulta em Clusters de Bancos de Dados", Tese de Doutorado, Programa de Engenharia de Sistemas e Computação, COPPE, Universidade Federal do Rio de Janeiro, Rio de Janeiro, Brazil.

[6] POSTGRESQL, "PostgreSQL DBMS", url: http://www.postgresql.org

[7] TPC (2003), "TPC Benchmark™ H – Revision 2.1.0", url: http://www.tpc.org.