ParGRES: *Middleware* para Processamento Paralelo de Consultas OLAP em Clusters de Banco de Dados

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

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

Abstract. *This paper describes ParGRES, a software for parallel processing of heavy-weight queries on top of database clusters. ParGRES is a middleware between the application and the database tiers that provides transparent access from the application to the parallel environment. Query processing in ParGRES combines intra- and inter-query parallelism techniques, while using database replication and virtual fragmentation. OLAP present typical heavy-weight queries and are one of the main targets of ParGRES.*

Resumo. *Este artigo descreve o ParGRES, um software que tem como objetivo o processamento paralelo de consultas de alto custo sobre clusters de banco de dados. ParGRES funciona como um middleware entre a camada da aplicação OLAP e o banco de dados, tornando o processamento paralelo transparente à aplicação. O processamento das consultas explora o paralelismo intra- e inter-consultas, usando replicação e fragmentação virtual de dados. Consultas OLAP são típicas de alto custo e um dos alvos principais do ParGRES.*

1. Introdução

O aspecto de desempenho de sistemas de informação tem se tornado cada vez mais crítico nas organizações, em função do crescente volume de dados manipulado pelos sistemas e da complexidade das requisições feitas à base de dados. Em especial, a complexidade no acesso aos dados armazenados tem grande influência em aplicações OLAP (*On-Line Analytical Processing*), que acessam grandes conjuntos de dados através de consultas de alto custo, de natureza não previsível (*ad-hoc*) [Gorla, 2003], e voltadas para análises de nível gerencial. Neste tipo de aplicação, requisições de atualização de dados são menos freqüentes [TPC, 2003]. A otimização de bancos de dados para suporte a consultas *ad-hoc* é mais complexa, uma vez que elas não são predefinidas. 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.

Tradicionalmente, o problema de desempenho de aplicações tem sido endereçado através de processamento paralelo, com a substituição da plataforma de hardware e software por componentes de maior capacidade computacional (como servidores e SGBDs paralelos) e a correspondente adaptação da aplicação do ambiente seqüencial para o ambiente paralelo. Nesta situação, a migração de uma aplicação é bastante complexa, e muitas vezes inviável, uma vez que pode requerer alterações no código-fonte. Além disso,

esta solução representa alto custo para a organização, tanto para a aquisição/expansão do ambiente computacional quanto para a migração da aplicação. Uma alternativa mais barata é a utilização de clusters de PCs. Porém, os custos de processamento de grandes massas de dados são muito altos, mesmo para clusters, seja pelo próprio software (SGBD) ou pelo hardware específico requisitado, como unidades centrais de armazenamento (SAN – *Storage Area Network*).

Outra proposta existente na literatura para exploração de processamento paralelo em aplicações sobre bancos de dados é denominada "cluster de bancos de dados" [Röhm *et al.*, 2002], que utiliza 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 o alto custo de migração das aplicações e dos bancos de dados, existente nos SGBDs paralelos.

O software descrito neste artigo segue a abordagem de cluster de banco de dados, e está sendo desenvolvida dentro do projeto de pesquisa ParGRES [Mattoso *et al.*, 2005], financiado pela Finep e Itautec através do Edital de Software Livre CT-INFO - 01/2003.

ParGRES é uma camada de software entre a aplicação e os SGBDs, capaz de coordenar as operações de acesso aos dados para obter o paralelismo desejado utilizando um cluster de PCs padrão. O paralelismo do ParGRES é voltado para consultas SQL que consomem muito tempo de processamento, como por exemplo, as consultas típicas de OLAP. Propomos uma solução que combina diferentes técnicas de processamento paralelo de consultas em um cluster de BD, além de tratar requisições de atualização de dados e o balanceamento de carga entre os nós do sistema. Na implementação atual, o SGBD utilizado em cada nó é o PostgreSQL [PostgreSQL, 2005]. No entanto, a comunicação entre o ParGRES e o SGBD é baseada no padrão SQL, não sendo dependente de nenhuma característica específica do SGBD. Isto flexibiliza a escolha pela organização do SGBD a ser utilizado, e torna possível a utilização de SGBDs heterogêneos.

Este trabalho está organizado conforme a seguir. A seção 2 descreve a arquitetura do ParGRES. A seção 3 detalha o ambiente computacional em que o ParGRES foi implementado, enquanto a seção 4 analisa alguns trabalhos relacionados. Finalmente, a seção 5 conclui este trabalho.

2. Arquitetura do Sistema

Como as demais soluções para clusters de BD, o ParGRES consiste em uma camada intermediária de software (*middleware*) que orquestra instâncias de SGBDs em execução paralela nos diferentes nós do cluster. Em vários projetos, como o C-JDBC [Cecchet *et al.*, 2004] e o PowerDB [Röhm *et al.*, 2002], essa camada é centralizada e executada em um nó escolhido como coordenador. O ParGRES, no entanto, possui arquitetura descentralizada (Figura 1). Seus componentes se encontram distribuídos entre os nós do cluster. A arquitetura foi baseada em [Lima, 2004].

Há componentes globais e locais. Componentes globais 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.

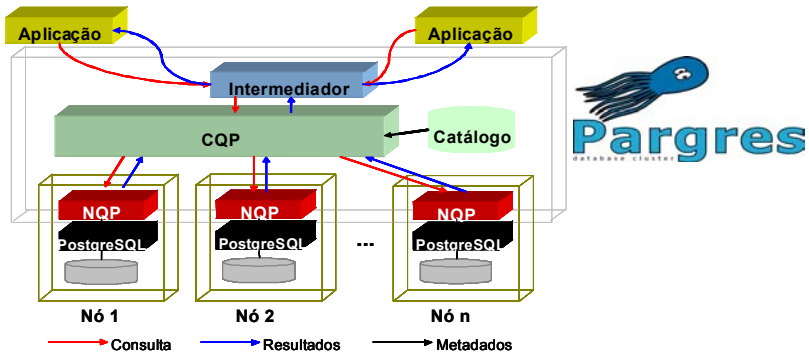

Figura 1 – Arquitetura do ParGRES

O componente mais importante é o CQP, que atua como coordenador de todos os demais e 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, o componente Intermediador tem a função de 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 a cada requisição, o que aumenta o grau de tolerância a falhas do ParGRES. Além disso, essa estratégia prepara a arquitetura para uma futura implementação distribuída do CQP.

Há quatro tipos de tarefas executadas pelo ParGRES: (i) tradução da consulta SQL para permitir a execução paralela, (ii) processamento de consultas com paralelismo inter/intra-consultas, (iii) composição de resultados e (iv) processamento de atualização.

O CQP possui um Tradutor que contém um analisador sintático para processar requisições de uma aplicação cliente em SQL-99 utilizando uma gramática livre de contexto apropriada. Gera como saída um conjunto de objetos que armazenam informações necessárias para as outras etapas do processamento. As principais informações geradas pelo tradutor são: (i) identificação das relações e atributos presentes na consulta que possam ser utilizados no paralelismo intra-consulta, (ii) informações para o CQP compor o resultado, e (iii) identificação dos atributos utilizados em agregações. Após o Tradutor identificar as relações e atributos, o CQP decide qual atributo será utilizado na fragmentação e solicita ao Tradutor uma versão da consulta apropriada à fragmentação virtual. Essa versão da consulta irá conter um predicado que determina as faixas da fragmentação virtual.

O CQP é 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 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, mantendo a filosofia de se utilizar o SGBD como um componente do tipo "caixa-preta".

O paralelismo intra-consulta (**intra-c**) significa decompor consultas complexas em sub-consultas que serão executadas em paralelo, cada sub-consulta 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. No paralelismo inter-consultas (**inter-**

c), consultas distintas são executadas de forma concorrente no cluster de BD, uma em cada nó do cluster.

No paralelismo inter-c, o CQP apenas envia a consulta ao NQP do nó com menor número de tarefas pendentes. 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.

Para prover alto desempenho para o processamento de consultas de alto custo, o ParGRES implementa o paralelismo intra-c utilizando a técnica de fragmentação virtual adaptativa [Lima *et al.*, 2004] com replicação total de dados. Como várias consultas podem ser processadas simultaneamente no cluster, inclusive pelos mesmos nós, o paralelismo inter-c é empregado juntamente com o intra-c. Uma das vantagens dessa combinação é que algumas consultas podem ser de baixo custo, não justificando a utilização de paralelismo intra-c, bastando a utilização de paralelismo inter-c.

A fragmentação virtual adaptativa, empregada pelo ParGRES, depende da existência de um índice *clustered* sobre uma das relações envolvidas na consulta para ser executada com sucesso [Lima *et al.*, 2004]. Esse tipo de informação está presente no Catálogo. Ele armazena os nomes e cardinalidades das relações que possuem índices *clustered*, atributo(s) sobre o(s) qual(is) cada índice é baseado e o intervalo de valores de cada um desses atributos. Como cada relação pode possuir apenas um índice *clustered*, o número de informações não é muito elevado.

No paralelismo intra-c, o CQP aloca os NQPs e envia a cada um deles um plano de execução local para a consulta. Cada NQP processa 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.

O ParGRES realiza a composição de resultados adaptando o algoritmo de agregação em duas fases [Shatdal e Naughton, 1995]. Nosso algoritmo utiliza processamento paralelo nesta composição minimizando a comunicação entre os nós. Na primeira fase, os nós agregam os grupos retornados pelas sub-consultas realizadas localmente. Na fase dois, os grupos são distribuídos para os seus respectivos nós através de uma função *hash*. Por fim, cada nó envia seu sub-conjunto do resultado global ao nó coordenador, que executa a sua união. Um destaque pode ser dado para a ordenação, que necessita de uma fase extra também executada em paralelo pelos nós do cluster.

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-c 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, o ParGRES adota uma política conservadora e não permite execução paralela de atualizações e consultas. Além disso, enquanto as atualizações são rápidas, as consultas possuem um tempo de processamento significativo [TPC, 2003]. 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.

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 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 [Lima, 2004]. Os resultados apresentados no referido trabalho mostram se tratar de uma técnica bastante eficiente, especialmente em situações de extrema desigualdade de cargas iniciais.

3. Ambiente computacional

A utilização do ParGRES pressupõe um ambiente computacional com uma arquitetura em 3 camadas: a aplicação (ferramenta OLAP), o ParGRES e os SGBDs instalados nos nós do cluster de PCs. Cada SGBD acessa sua base de dados local, em que os cubos de dados já estejam gerados e prontos para serem consultados pela ferramenta OLAP. Neste cenário, supõe-se que a ferramenta OLAP obtenha os dados através de consultas SQL.

A comunicação do ParGRES com a ferramenta OLAP é simples, bastando redirecionar a configuração do servidor de BD acessado pela ferramenta para o cluster em que o ParGRES esteja executando. As consultas OLAP passam a ser interceptadas pelo CQP, que será responsável pelo seu processamento paralelo, de forma transparente à aplicação. O Tradutor foi escrito usando a ferramenta BYacc/J, que é o tradicional Yacc com adaptações para gerar código em Java.

ParGRES está sendo desenvolvido em Java, e os testes estão sendo executados em um cluster de PCs seguindo arquitetura de memória distribuída. Pela sua arquitetura, o ParGRES não necessita em absoluto de hardware específico de clusters, tais como unidades SAN ou placas de redes de alta velocidade. Na implementação atual do ParGRES está sendo utilizado o SGBD PostgreSQL versão 8.0 em cada nó do cluster. A comunicação entre o ParGRES e cada nó do cluster é realizada através de JDBC.

4. Trabalhos relacionados

As principais soluções de clusters de BD existentes na literatura são C-JDBC [Cecchet *et al.*, 2004] e PowerDB [Röhm *et al.*, 2002]. 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 várias 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. Nesse caso, a solução inter-consultas não é atraente, pois não reduz esse tempo. O ParGRES é fortemente baseado nos algoritmos de paralelização propostos em [Lima, 2004], porém acrescenta a tradução automática da consulta SQL e

permite operações de atualização, tornando-o uma solução que facilita a migração de aplicações seqüenciais OLAP para a solução paralela em clusters de PC. Na linha de software livre, o Mondrian [Mondrian, 2005] é um sistema voltado para análises OLAP. O Mondrian pode se beneficiar do paralelismo do ParGRES ao incluir o *middleware* entre a ferramenta OLAP e o SGBD que processa suas consultas, sendo assim um trabalho complementar ao nosso. Ainda na área de processamento OLAP, o projeto Panda [Chen *et al.*, 2004], dentre outros, endereça a geração em paralelo dos cubos de dados em ambientes paralelos. O ParGRES não é específico para consultas OLAP. Além disso, a ferramenta visa acelerar o desempenho de aplicações já em produção, assim a construção da base de dados ou do cubo está fora do escopo desse trabalho.

5. Conclusão

O ParGRES é um software livre que provê paralelismo para a execução eficiente de aplicações OLAP sobre clusters de BD. O ParGRES implementa o paralelismo intra- e inter-consultas, e usa a técnica de fragmentação virtual adaptativa com replicação total de dados, o que torna dispensável refazer o projeto físico da base de dados quando da migração para o ambiente distribuído. Por este motivo, e também por usar SGBDs "caixa-preta" e não necessitar de hardware específico para seu funcionamento, o ParGRES se caracteriza como solução de baixo custo para o problema de desempenho de aplicações que acessam um grande volume de dados através de consultas complexas. A comunicação entre o ParGRES e o SGBD utiliza o padrão SQL, portanto qualquer SGBD que implemente o padrão pode ser adotado.

Embora a replicação total da base de dados possa ser apontada como ponto fraco da abordagem proposta, é importante observar que o custo da memória secundária vem caindo drasticamente, o que torna a abordagem hoje economicamente atraente mesmo para grandes bases de dados. Além disso, a replicação total beneficia o sistema através do aumento da disponibilidade e da tolerância a falhas.

Outras soluções existem baseadas em clusters de BD, no entanto o diferencial do ParGRES é prover paralelismo intra-consulta de modo transparente, reduzindo o tempo de resposta de cada consulta de alto custo enviada pela aplicação.

6. Referências

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

Chen, Y., Dehne, F., Eavis, T., Rau-Chaplin, A. (2004), "Parallel ROLAP Datacube Construction on Shared Nothing Multi-Processors", Journal of Parallel and Distributed Databases, 15 (3), pp. 219-236.

Gorla, N. (2003), "Features to Consider in a Data Warehousing System",Comm ACM,46(11),pp. 111-115.

Lima, A. A. B., Mattoso, M. and Valduriez, P. (2004), "Adaptive Virtual Partitioning for OLAP Query Processing in a Database Cluster", In: Proc 19th SBBD, Brasília, Brasil, pp. 92-105.

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, UFRJ, Brasil.

Mattoso, M., Zimbrão, G., Lima, A. A. B., Baião, F., Braganholo, V., Aveleda, A., Miranda, B., Almentero, B., Costa, M. (2005), "ParGRES: uma camada de processamento paralelo de consultas sobre o PostgreSQL". In: Proc WSL - Workshop de Software Livre, Porto Alegre, pp. 259-264.

Mondrian (2005), "Mondrian OLAP Server", url: http://mondrian.sourceforge.net/, acesso em Jul/2005

PostgreSQL (2005), "PostgreSQL v.8.0", url: http://www.postgresql.org/download/, acesso em Jul 2005.

Röhm, U., Böhm, K., Schek, H.-J., et al. (2002), FAS - A Freshness-Sensitive Coordination Middleware for a Cluster of OLAP Components, Intl. Conf. Very Large Databases, Hong Kong, pp. 754-765.

Shatdal, A., Naughton, J. (1995), "Adaptive Parallel Aggregation Algorithms", SIGMOD, pp.104-114.

TPC (2003), "TPC BenchmarkTM H – Revision 2.1.0", url: http://www.tpc.org.