PARGOA: Um Servidor Paralelo para Sistemas de Banco de Dados Orientados a Objetos

Marta L. Queirós Mattoso Jano Moreira de Souza Claudio Luis de Amorim

Programa de Engenharia de Sistemas e Computação
COPPE/UFRJ
Caixa Postal 68511, Rio de Janeiro, RJ, 21945-970
e-eletrônico: marta, jano, amorim @cos.ufrj.br

Sumário

Este trabalho, apresenta uma estratégia de paralelismo para Sistemas de Gerência de Bases de Dados Orientados a Objetos (SGBDOOs). Esta estratégia, baseada no modelo de memória de disco compartilhado, foi implementada em um servidor paralelo de objetos, o PARGOA. Medições de desempenho efetuadas na máquina paralela NCP I confirmam a adequação dos procedimentos desenvolvidos para a manipulação de objetos armazenados.

Abstract

This work presents a parallel processing strategy for object oriented database management systems. This strategy is based on the shared disk memory model and was implemented on the PARGOA, a parallel object server. Performance measurements over the parallel machine NCP I show the effectiveness of the proposed solution.

1 INTRODUÇÃO

A experiência das máquinas paralelas de banco de dados e dos diversos protótipos apresentados em levantamentos na literatura [Matt91b, DeWi92] vem mostrando que o modelo relacional é naturalmente paralelizável e que o ganho em desempenho que vem sendo atingido é hoje inquestionável. Num Sistema de Gerência de Bases de Dados (SGBD) relacional, é possível distinguir três fontes de paralelismo. O paralelismo explorado entre transações de diversos usuários, o paralelismo entre as operações que compõem uma transação e o paralelismo dentro de uma operação. O paralelismo intra-operação é o mais explorado e é atingido através de algoritmos paralelos que tiram partido do particionamento dos dados.

No caso do modelo de dados orientado a objetos, o paralelismo não é imediato, pois as características dos dados do modelo relacional diferem da orientação a objetos em diversos aspectos. A grande fonte de paralelismo do modelo relacional se apoia no processamento de conjuntos **uniformes** de dados, através de comandos predefinidos, por exemplo, a linguagem de manipulação de dados. Já na orientação a objetos, os objetos pertecem a classes mas não necessariamente caracterizam um conjunto. Objetos são manipulados por métodos e por alguns "comandos" existentes na interface de acesso e de manipulação de objetos. Além disso, existe uma preocupação na disposição dos objetos no disco de modo a **agrupá-los** de acordo com o melhor padrão de acesso. Embora uma das técnicas de agrupamento seja juntar fisicamente todos os objetos de uma classe, existem outras técnicas que privilegiam o relacionamento de composição entre os objetos, armazenando junto, objetos de classes diferentes.

Segundo DeWitt e Gray [DeWi90a] o agrupamento de objetos é ainda um tema de pesquisa em aberto e a combinação com a distribuição de objetos para permitir o paralelismo torna o problema ainda mais complexo. Entretanto, esses aspectos não inviabilizam a exploração de paralelismo em Sistemas de Gerência de Bases de Dados Orientados a Objetos (SGBDOOs), mas apenas mostram que técnicas de paralelismo utilizadas em SGBDs relacionais não possuem aplicabilidade direta com os modelos orientados a objetos. Esses fatores levaram a pesquisas de novas fontes a serem exploradas dentro das arquiteturas de SGBDOOs.

A busca de desempenho através do paralelismo levou as arquiteturas de SGBDOOs a adotarem uma plataforma cliente/servidor [DeWi90b], onde parte do código do SGBDOO fica no cliente e parte no servidor. O cliente é executado em um processador e o servidor em outro. Com essa distribuição, obtém-se um paralelismo entre a execução dos códigos do cliente e do servidor. Outra fonte de paralelismo que pode ser explorada nos SGBDOOs está na execução das operações predefinidas, ou seja, operações da linguagem de manipulação de objetos. Como exemplo, podem ser citados os comandos da linguagem de consulta e alguns comandos de manipulação. O paralelismo dessas operações pode ser entre operações de usuários e dentro de operações de um único usuário. Entretanto, torna-se necessária a investigação sobre os conflitos entre agrupamento e particionamento para que o paralelismo possa ser explorado efetivamente no modelo orientado a objetos.

Este trabalho, apresenta uma estratégia de paralelismo para SGBDOOs. Esta estratégia, baseada no modelo de memória de disco compartilhado, foi implementada em um servidor paralelo de objetos, o PARGOA. Medições de desempenho efetuadas na máquina paralela NCP I [Amor91] confirmam a adequação dos procedimentos desenvolvidos para a manipulação de

objetos armazenados. Nesse sentido, a Seção 2 apresenta um panorama geral sobre fontes de paralelismo na orientação a objetos e suas dificuldades de implementação. Nessa Seção, também são abordadas as soluções encontradas na literatura que, no entanto, não contemplam o modelo de dados orientado a objetos. Na Seção 3 são discutidas as características da arquitetura paralela com modelo de memória de disco compartilhado e suas implicações na solução adotada para este trabalho. Já a Seção 4 apresenta o protótipo do servidor paralelo desenvolvido com suas características e avaliação de desempenho e, finalmente, a Seção 5 apresenta comentários finais sobre a experiência realizada.

2 O PARALELISMO E A ORIENTAÇÃO A OBJETOS

Embora a utilização de estratégias paralelas na orientação a objetos apresente uma série de dificuldades, o paralelismo pode ser utilizado junto às aplicações não convencionais de SGBDOOs que necessitam de desempenho. A exploração de paralelismo num SGBDOO pode ocorrer em diversos níveis. Seja de uma forma geral no SGBDOO, através da utilização de uma linguagem de programação paralela e orientada a objetos, por exemplo, ou em operações delimitadas dentro do SGBDOO. Entretanto, segundo DeWitt [DeWi90a], os pontos mais críticos estão na exploração de paralelismo nos métodos do usuário e na conciliação da distribuição dos objetos com o agrupamento adequado à gerência de objetos.

2.1 O Paralelismo nas Operações e Métodos do SGBDOO

A exploração do paralelismo dentro dos métodos escritos pelo usuário ainda é um tema em aberto, pois é dependente da linguagem utilizada. Entretanto, três opções podem ser vislumbradas [DeWi90a, Matt93]. Na primeira opção, se o usuário faz uso da linguagem de programação própria do banco de dados ou utiliza os comandos da linguagem de consulta, o otimizador, junto ao executor dos comandos, poderá tirar proveito de uma execução paralela de modo transparente ao usuário. Outra opção é fazer com que o usuário programe seus métodos em uma linguagem de programação paralela. Numa terceira opção, o paralelismo do método seria explicitado pelo usuário. Caso seja utilizada uma linguagem de programação convencional, seria necessária a utilização de um pré-processador que identificaria comandos do tipo 'parallel do'.

Dentro de uma proposta mais ambiciosa, poderia ser utilizada uma linguagem de programação paralela orientada a objetos para a programação dos métodos. Diversos autores [Jézé92, Agha86] apontam o mundo de objetos como sendo altamente propício ao paralelismo e concorrência, devido ao encapsulamento dos objetos. Entretanto, esse mundo não é tão autônomo assim, uma vez que no mínimo o objeto pertence a uma hierarquia de classes, se relacionando implicitamente, com outros objetos. A dificuldade da gerência desta hierarquia torna limitada a maioria dessas linguagens [Wyat92]. Por exemplo, POOL2 [Amer91] não permite a herança de classes no modelo de objetos da linguagem.

2.2 A Questão do Agrupamento X Distribuição

Como na orientação a objetos o conceito de classes não coincide com o de conjuntos, é necessário que se escolha sobre que grupo de objetos será realizado a distribuição dos dados. Segundo DeWitt [DeWi90a], é preciso optar entre a distribuição dos objetos de todas as classes e

entre a distribuição de classes do tipo *coleção* somente. Além disso, deve ser considerado a distribuição de coleções de objetos que são referenciadas em atributos multi-valorados.

Uma vez identificadas as classes ou coleções que terão seus objetos distribuídos através de alguma técnica de fragmentação, surge a questão sobre como conciliar a indicação de **particionamento** de uma coleção para o processamento paralelo, com a indicação conflitante de **agrupamento** para o processamento em conjunto. Outro problema que surge neste conflito está em que fazer quando os objetos referenciados são armazenados junto ao objeto raiz da hierarquia de composição, ao mesmo tempo que os objetos referenciados pertencem a uma coleção.

2.3 As Soluções Anteriores

Poucas são as propostas encontradas na literatura para exploração de paralelismo em SGBDOOs. Existem projetos que apresentam soluções no âmbito dos chamados bancos de dados de terceira geração [Ston90] que possuem a proposta de suportar aplicações não convencionais através de extensões ao modelo relacional. Como exemplo, podem ser citados os projetos/protótipos EDS [Vald90, Sala91], PRIMA [Mits92], XPRS [Ston88, Hong92] e VOLCANO [Grae90].

Nestes sistemas, a aplicação pode ser modelada com estruturas mais ricas que o modelo relacional, permitindo uma modelagem com semântica através de objetos complexos, entre outras características. Entretanto, o modelo interno adotado ainda é baseado em estruturas e operações da álgebra relacional. Como o paralelismo nestes sistemas é empregado apenas no nível interno do processamento das operações, as técnicas de paralelismo utilizadas nos Sistemas de Banco de Dados Paralelos (SBDPs) relacionais são quase que diretamente aplicáveis.

O projeto EDS, utiliza uma extensão da linguagem SQL (ESQL) [Berg91] para a manipulação dos objetos que é mapeada para uma extensão da álgebra relacional (LERA). A arquitetura do sistema EDS adota a solução de cliente/servidor com o servidor paralelo de dados com capacidade de processar operações da álgebra relacional. No projeto XPRS, também é realizado o paralelismo nas operações da álgebra e pretende-se que o processamento paralelo seja integrado ao sistema PostGres [Ston90]. Já no projeto PRIMA, é utilizado um modelo de dados próprio, o MAD que também é mapeado para extensões da álgebra relacional. A arquitetura do PRIMA, também utiliza a divisão cliente/servidor de objetos, onde o servidor de objetos deverá ser executado sobre uma máquina paralela de memória compartilhada da Sequent.

Esses projetos, limitam o paralelismo às operações realizadas no contexto do servidor de dados e especificamente no processamento de consultas. Por outro lado, pode ser observado que a maioria das soluções existentes utiliza uma proposta conservadora, fazendo uso do paralelismo num servidor de dados **relacional** que se comunica com clientes, onde os dados são mapeados para um modelo mais rico em semântica.

3 A SOLUÇÃO ADOTADA NA GERÊNCIA PARALELA DE OBJETOS

Para solucionar os problemas de paralelismo na orientação a objetos apresentados na Seção 2, foram adotadas as seguintes estratégias:

i) Optou-se pela utilização da arquitetura de **disco compartilhado** (DC) no sentido de conciliar os requisitos de agrupamento com o particionamento.

ii) Para atacar o problema de paralelismo sobre métodos, foi limitado o escopo das operações paralelizáveis através da utilização de um **servidor paralelo de dados** [Vald90a] que privilegia as operações sobre conjuntos de objetos. Nesse sentido, os métodos poderão usufruir de paralelismo quando utilizarem as operações predefinidas do servidor de objetos, como por exemplo, as operações de consulta.

3.1 A Utilização da Arquitetura de Disco Compartilhado

Não existe um consenso quanto ao melhor modelo de memória a ser adotado em uma arquitetura de banco de dados, e diversos trabalhos [Bhid88, Laks89, DeWi90c, DeWi92] dividem-se favoravelmente a um dos três modelos classificados por [Ston86] como arquiteturas de memória totalmente compartilhada (MC), memória totalmente distribuída (MD) e memória de disco compartilhado (DC). Essa classificação, visa caracterizar o acesso dos processadores à memória principal e secundária, onde o modelo DC é o intermediário, tendo a memória principal distribuída e a memória secundária compartilhada por todos os processadores.

Para tentar resolver a questão da dicotomia entre o particionamento e o agrupamento de objetos, foi adotada a solução conciliatória da arquitetura paralela de disco compartilhado. Nesta arquitetura, ao contrário da memória distribuída, os dados/objetos não precisam estar fisicamente distribuídos pelos discos e alocados aos processadores. Cabe ao Gerente de Distribuição escolher a melhor maneira de "enviar" os objetos aos processadores. Desta forma, o processamento seqüencial mantém o mesmo padrão de acesso aos objetos em disco e o processamento paralelo realiza a distribuição dinamicamente no momento da execução da operação paralelizável. Sendo assim, os objetos possuem uma disposição no disco não comprometida com o acesso paralelo às coleções, mas sim com a melhor estratégia de acesso à base de um modo global.

3.2 O Servidor Paralelo de Objetos na Arquitetura de um SGBDOO

Devido aos diversos problemas encontrados no particionamento dos dados orientados a objetos e principalmente devido à dificuldade de execução de métodos em paralelo, surgiu a opção de aplicar o processamento paralelo no escopo de um servidor de objetos. Num servidor de objetos, dentro de uma arquitetura de SGBDOO, a unidade de transferência entre o servidor e o cliente é o objeto. O servidor conhece o conceito de objeto e é capaz de executar consultas e métodos sobre os objetos.

A utilização de paralelismo numa arquitetura de SGBDOO no escopo do servidor de objetos, limita as operações a serem paralelizadas, porém engloba o processamento de consultas que ainda é uma das maiores fontes de paralelismo. Entretanto, o uso da orientação a objetos em SGBDs ainda não atingiu maturidade científica, e a incorporação de uma máquina paralela para hospedar todo o SGBDOO implicaria numa complexidade difícil de ser gerenciada. Desta forma,

optou-se por uma solução onde o paralelismo fica direcionado às operações do servidor de objetos, com ênfase na manipulação de coleções de objetos. A utilização do paralelismo no contexto do servidor é também adotada nos sistemas PRIMA, EDS e VOLCANO.

A Figura 1 mostra a proposta do servidor paralelo de objetos no âmbito do SGBDOO GEOTABA [Matt93]. A proposta de paralelismo na arquitetura do GEOTABA consiste, então, da substituição do servidor de objetos GOA (Gerente de Objetos Armazenados) da estação de trabalho seqüencial, por um servidor paralelo GOA instalado sobre uma máquina paralela, o NCP I. O cliente do GEOTABA fica responsável pelas gerencias do esquema de objetos, da troca de mensagens entre os objetos, da interface com usuário, de transações e da memória de objetos. O servidor gerencia o armazenamento de objetos, as transações e o processamento de consultas.

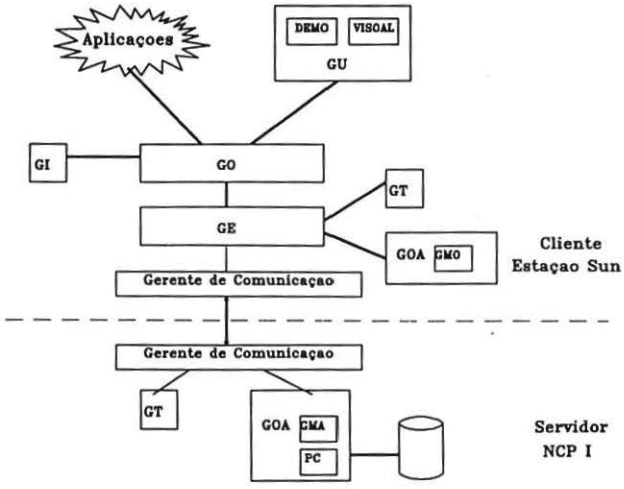

Figura 1 - Arquitetura cliente/servidor paralelo do GEOTABA

As operações que o cliente solicita ao servidor paralelo GOA são exatamente as mesmas do servidor seqüencial, ou seja, armazenamento e recuperação de objetos, gerência de coleções e avaliação de predicados de consulta. Entretanto, antes da execução, existe um controle no servidor paralelo que analisa que nível de granularidade pode ser aplicado à operação. Caso a operação possa ser distribuída entre os diversos processadores, o paralelismo é *dentro da operação* e a operação é executada em paralelo. Caso a operação não seja intra-paralelizável, como por exemplo, a solicitação do cliente para que lhe seja enviado um determinado objeto, o seu processamento é realizado em um único processador deixando os demais processadores livres para a execução de outras operações. Ao término da execução da operação, o servidor paralelo envia o resultado para o cliente.

Estudos de DeWitt e outros [DeWi90b] demonstram que a opção do servidor de objetos é a menos sensível ao agrupamento de objetos no disco, já que a memória do cliente só contém os

dados necessários ao cliente. Essa característica reforça ainda mais a opção pelo servidor de objetos paralelo. Como desvantagem, o servidor de objetos limita suas fontes de paralelismo na medida em que informações do esquema são gerenciadas pelo cliente somente. O compartilhamento de informações do esquema entre cliente e servidor implicaria em um aumento considerável da complexidade do servidor, principalmente por conta do controle da redundância. Nesse sentido, esta versão do servidor paralelo fica limitada às informações do esquema do modelo interno de representação, disponíveis no servidor.

4 A IMPLEMENTAÇÃO DO SERVIDOR PARALELO DE OBJETOS

Para avaliar a proposta de paralelismo em SGBDOOs proposta na Seção anterior, foi desenvolvido o PARGOA que é um protótipo de servidor paralelo de objetos. A implementação deste protótipo une as experiências obtidas a partir do desenvolvimento do PARBASE [Matt91a], um gerente paralelo de operações da álgebra relacional, e do servidor de objetos GOA [Matt93]. O resultado foi uma mistura de módulos dos protótipos já desenvolvidos, onde parte do código do PARBASE foi reutilizado para o Gerente de Distribuição do PARGOA e as operações seqüenciais dos nós consistem de pequenas adaptações ao código do servidor do GOA. Além das vantagens da utilização do mesmo ambiente do PARBASE, versões mais recentes do NCP I [Amor91] contam com uma série de atrativos extras descritos a seguir.

4.1 O Ambiente de Desenvolvimento

Devido a testes ainda em realização sobre o computador paralelo NCP I, que conta em cada nó com um transputer e um microprocessador i860, a implementação do PARGOA ficou limitada ao modelo T8 do NCP I, onde cada nó contém um transputer e 2 Mbytes de memória. O computador hospedeiro utilizado foi um PC-AT, entretanto, o disco rígido utilizado foi de capacidade inferior ao modelo T8 das experiências do PARBASE. Entretanto, no protótipo do NCP I em experimentação e não aberto ainda aos usuários, já está instalado, dentro do modelo de memória de disco compartilhado, um disco rígido com capacidade de armazenamento de 676 Mbytes de informação, acessado pelos 8 processadores através da interface SCSI - 'Small Computer System Interface' que viabiliza o acesso concorrente. Esse disco é gerenciado pelo sistema operacional Helios, possui um 'cache' próprio. Os testes realizados com o disco e a nova interface SCSI apontaram um ganho na velocidade de transmissão de até 40 vezes mais rápido em relação à utilização do disco rígido com controladora MFM, atualmente no PC. Além disso, os nós já contam com uma memória local com 4 Mbytes de armazenamento (o dobro da capacidade atual e utilizada nos testes do PARGOA).

Quanto ao **ambiente de programação**, foi utilizada a linguagem Strand [Arti90] para as rotinas de gerência do paralelismo e foi utilizada a linguagem C para o código seqüencial carregado nos processadores. Na realidade, o código carregado nos nós é exatamente igual ao código utilizado pelo servidor seqüencial do GOA nas estações Sun. O código foi totalmente compatível com o C do sistema operacional Helios do NCP I. Conforme esperado, não foi necessária nenhuma modificação para o transporte do código. A característica do STRAND de permitir a associação direta de código a processadores, aliado à sua interface para a linguagem C, fizeram com que qualquer operação do servidor seqüencial do GOA pudesse ser executada por qualquer processador do NCP I.

4.2 Características do PARGOA

Apesar do PARGOA oferecer diversas oportunidades de paralelismo em sua execução, algumas limitações foram realizadas na implementação corrente, pois o objetivo principal do protótipo foi de analisar o grau de dificuldade ou facilidade na exploração do paralelismo. Foi realizado, antes de mais nada, o estudo de viabilidade da utilização de processamento paralelo no contexto da gerência de dados com orientação a objetos. A seguir são apresentadas as opções realizadas quanto às técnicas de paralelismo escolhidas no projeto do PARGOA.

i) Nível de granularidade

No nível de granularidade do processamento paralelo de operações, o PARGOA permite que os três níveis apresentados na literatura[Öszu, Matt91b] sejam explorados. O paralelismo *dentro de operações* é explorado por meio das operações de avaliação de predicados do servidor através da distribuição dos objetos da coleção alvo. Já no nível *entre operações*, pode ser citado a realização do armazenamento paralelo de dois objetos de classes diferentes, ou da modificação de um objeto e da remoção de outro, entre outras operações.

Embora o paralelismo *entre transações*, possa ser explorado, as aplicações dos SGBDOOs em geral possuem transações longas e a gerência desse paralelismo torna-se mais difícil. Além disso, a transação longa pode envolver diversas operações gráficas de interface com o usuário e a execução de métodos programados pelo usuário. Tais características fazem com que se desconheça as operações sendo realizadas, para que o paralelismo possa ser explorado, a menos que seja explicitado pelo usuário ou que utilize operações predefinidas.

ii) Modelo de memória

Conforme já exposto na Seção 3, a implementação do PARGOA tira proveito da arquitetura de disco compartilhado do NCP I que não impõe uma distribuição física dos objetos pelos processadores. Cada processador dispõe de memória local para o 'buffer' de páginas e tem acesso ao mesmo espaço em disco.

iii) Fragmentação das coleções

Devido aos problemas apresentados quanto à dificuldade de conciliar o agrupamento da orientação a objetos com o particionamento do processamento paralelo, várias estratégias foram avaliadas. O ponto de partida da implementação foi não prejudicar as técnicas de agrupamento adotadas no servidor de objetos seqüencial do GOA. Para escolher a técnica de distribuição, as características do modelo interno e agrupamento dos objetos tiveram que ser consideradas. Como por exemplo:

(a) O escopo da coleção alvo, contém somente os objetos inseridos na coleção e não necessariamente todos os objetos da classe associada à coleção alvo.

(b) O acesso à extensão de uma classe não está disponível, nem mesmo a informação de que essa classe seja do tipo coleção.

⇨ Os objetos de classes do tipo coleção são preferencialmente armazenados em segmentos de tamanho fixo exclusivos para o armazenamento da coleção. À medida que um segmento fica cheio, outros vão sendo criados.

⇨ Já objetos complexos compostos podem ter indicação para serem armazenados junto com seus sub-objetos, mesmo que estes pertençam a uma coleção.

(c) os objetos de uma coleção estarão quase sempre armazenados contiguamente.

Como não existe um comportamento/armazenamento uniforme entre os objetos das classes definidas na base, a estratégia de distribuição dos objetos e alocação dos processadores tenta conciliar dois objetivos, a saber: 1) minimizar o número de acessos a disco tirando proveito da estratégia de armazenamento, e 2) minimizar o acesso concorrente a páginas compartilhadas no disco.

Nesse sentido, a fragmentação só ocorre no processamento de coleções. As coleções são distribuídas através da fragmentação **horizontal,** por analogia ao modelo relacional, onde **objetos** (tuplas), e não atributos, são distribuídos pelos processadores. O desagrupamento no PARGOA só é total quando a coleção utiliza **n** ou mais segmentos, onde **n** equivale ao número de processadores da configuração da máquina paralela. Foi adotado o **desagrupamento parcial,** uma vez que os resultados obtidos com o PARBASE [Matt91a] e trabalhos analíticos [Bora88], apontam diversas vantagens sobre o desagrupamento total, que no entanto é mais fácil de ser implementado.

iv) Política de Distribuição

A política de distribuição de objetos de uma coleção é uma variação da técnica circular ('round robin'). No momento da execução de uma operação de consulta, os segmentos que armazenam fisicamente os objetos de uma coleção são distribuídos entre os processadores, sem levar em conta os valores dos objetos. Foram avaliadas as opções de utilização de uma distribuição por faixa de valores, por exemplo, através do uso de índices parciais [Ston88]. Entretanto, a limitação das informações semânticas conhecidas pelo servidor de objetos, torna complexa a manutenção desses índices. Por exemplo, quando um objeto é modificado no cliente, o servidor não possui a informação sobre que atributos foram alterados. Esse tipo de estratégia de distribuição será avaliado em versões futuras, uma vez que contribuem para o desempenho do SGBD através da localidade conhecida, seja para a faixa de valores ou seja pela função de 'hashing' utilizada.

4.3 A Implementação do PARGOA

Para implementar todas as características do PARGOA (descritas na Seção anterior), foram definidos os módulos apresentados na Figura 2. Esses módulos tiram proveito das implementações do PARBASE e do GOA seqüencial.

Assim como no servidor seqüencial do GOA, o cliente possui à sua disposição todas as operações descritas na Seção 3. Embora o protocolo de comunicação tenha mudado, o cliente requisita a execução de uma operação com o mesmo código intermediario especificado para o servidor seqüencial [Matt93]. Ao receber a operação codificada, o servidor envia para o **Gerente de Coleções**, responsável pela obtenção de algumas informações descritivas da coleção ou classe envolvida.

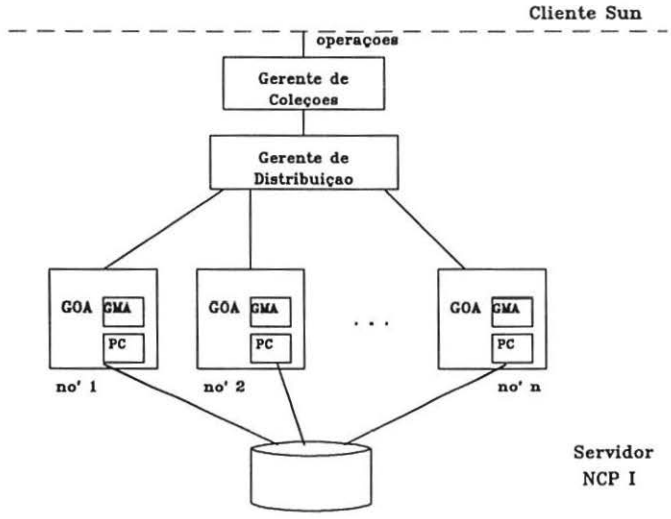

Figura 2 - Módulos do servidor paralelo de objetos do PARGOA

O **Gerente de Distribuição** é o responsável pela análise das características da operação para verificar se ela será executada em **1** ou **n** processadores e se é necessário a utilização de mecanismos de sincronização entre uma operação e outra, ou dentro da mesma operação. No caso do processamento de uma avaliação de predicado, o Gerente de Distribuição analisa a cardinalidade da coleção e o número de segmentos envolvidos. É enviado para cada processador, o código da operação e uma lista dos IDOs (identificadores dos objetos), local ao processador.

A implementação atual do PARGOA, não conta ainda com um controle de concorrência para garantir a consistência entre as páginas dos 'buffers' locais aos processadores. Desta forma, as operações paralelizáveis no PARGOA estão limitadas às operações que não envolvem gravações no disco, como é o caso do processamento de consultas do GOA. Pretende-se no futuro que o Gerente de Distribuição entre em sintonia com o Gerente de Transações para que seja adotada a técnica 'CLM - Central Lock Manager' [Bhid88] para implementar o controle centralizado para o acesso concorrente às páginas do disco.

O **Gerente de Objetos Armazenados** de cada processador é quase igual ao GOA seqüencial, já que nos nós o processamento é serial. A operação é identificada e o controle é desviado para o algoritmo responsável pela sua execução. Cada processador possui seu 'buffer' de páginas que é gerenciado com as mesmas rotinas do GOA seqüencial, entretanto, ao término da operação, os 'buffers' são liberados. Existe, no entanto, um tratamento de excessão para o buffer do nó 0, uma vez que o Gerente de Distribuição tem total controle sobre essa área. Neste caso, o buffer do nó 0 é gerenciado do mesmo modo que no GOA seqüencial e faz o papel do buffer de páginas dos clientes.

Foi dado um destaque ao módulo responsável pelo processamento de consultas no servidor paralelo, o PC, pois a avaliação de cláusulas de predicados no GOA paralelo é uma das grandes fontes de paralelismo explorada. O código seqüencial que fica carregado nos nós do NCP I, para realizar as operações de consulta, difere dos algoritmos do GOA seqüencial, na medida em que os processadores já recebem a lista de IDOs a ser percorrida e não a identificação da coleção como um todo. De modo análogo, as demais operações do GOA paralelo possuem pequenas variações em relação ao código seqüencial.

Embora a utilização do Gerente de Distribuição imponha uma centralização na gerência do paralelismo, ela é minimizada pelo modelo de disco compartilhado e pela topologia hipercúbica do NCP I. No modelo de disco compartilhado, uma vez tendo sido distribuídas as operações pelos nós, cada processador realiza sua comunicação com o disco em paralelo, sem envolver o Gerente de Distribuição. Na topologia do NCP I, o nó 0 intermedia a comunicação com o computador hospedeiro. Desta forma, o Gerente de Distribuição fica carregado no nó 0, por onde o código de chegada e saída do cliente teria que passar de qualquer maneira.

4.4 Análise das Operações de Consulta

O objetivo desta análise não é ser extensa ou completa através da medida de desempenho das operações de consulta, mas sim de mostrar a viabilidade do processamento paralelo em operações que utilizam um modelo de dados orientado a objetos. Foram realizados testes com avaliações de predicados através da estratégia descendente 'nested loops'. O Gerente de Distribuição analisa o número de objetos da coleção e opta por uma distribuição total ou parcial.

Na distribuição total, são distribuídos os segmentos que contém os objetos da coleção pelos processadores. Os objetos que não estão armazenados nos segmentos da coleção, são distribuídos de acordo com seu endereço físico. Os endereços são ordenados e divididos pelo número de processadores. Essa distribuição visa a obtenção de um bom balanceamento de carga dos objetos da coleção alvo entre os processadores, além de minimizar o acesso concorrente a páginas compartilhadas pelos processadores através de uso dos endereços físicos. Cada processador percorre sua lista de objetos penetrando nos níveis do acesso aos objetos das diversas classes envolvidas no caminho do predicado.

Exemplo: **select** Deptos **where** chefe.cidade.popul > 1 milhão.

Neste caso, são percorridas três classes, de acordo com a modelagem típica da aplicação Empresa, na seguinte ordem: Dept, Func e Cidade. A classe Dept está associada à coleção Deptos e é percorrida em paralelo. As classes Func e Cidade possuem os atributos cidade e popul respectivamente. Entretanto, o acesso aos objetos das classes Func e Cidade é concorrente e eventualmente compartilhado.

O **algoritmo descendente** de avaliação de predicados na arquitetura do PARGOA fica da seguinte maneira, para o exemplo da seleção de Deptos apresentado anteriormente:

Passo 1

- No Gerente de Distribuição (GD), os objetos da classe Dept são distribuídos pelos processadores com a cláusula "chefe.cidade.popul > 1 milhão".
- Nos processadores, os algoritmos locais aos nós avaliam e enviam para o GD os IDOs que qualificam a cláusula, através do mesmo algoritmo do servidor seqüencial.

A estratégia descendente nem sempre é a mais interessante devido ao compartilhamento no acesso às classes dos objetos referenciados encadeadamente. Caso as cláusulas da qualificação do predicado tivessem muitos níveis de objetos a serem percorridos, haveria grande probabilidade de se ter muita concorrência no acesso às páginas dos objetos. O acesso concorrente só não haveria nos objetos da coleção alvo, uma vez que estes foram desagrupados entre os processadores. Por outro lado, essa concorrência só é crítica no momento do acesso ao disco, já que nas consultas os objetos não são modificados e podem estar simultaneamente nos 'buffers' dos diversos processadores.

A seguir, são apresentados os resultados de uma avaliação do desempenho do protótipo do SBDP PARGOA na máquina paralela NCP I. Um sistema paralelo ideal apresenta duas propriedades chave [DeWi90a]: a aceleração linear e a expansibilidade linear. Foi realizada uma série de testes com predicados de seleção onde foram analisados os tempos de resposta medidos para variações no número de níveis de atributos da cláusula do predicado e no número de processadores utilizados na configuração da arquitetura. Os testes foram aplicados no sentido de medir-se a aceleração do PARGOA. As coleções utilizadas para o 'benchmark' são instâncias de uma base de dados real com objetos em média de 100 bytes. Foram realizados testes sobre coleções de 100 e 1000 objetos.

Todos os testes foram repetidos para configurações de 1, 2, 4 e 8 processadores no NCP I. Os tempos foram medidos a partir do momento que o cliente submete a operação ao PARGOA até a finalização da operação. Foram incluídas neste tempo todas as passagens da operação pelos diversos gerentes do PARGOA.

As Figuras 3 e 4, apresentam o ganho obtido com o aumento no número de processadores para alguns dos testes realizados. Cabe lembrar que o ganho é a razão entre o tempo de execução com **1** processador e o tempo de execução com **n** processadores. São apresentados os resultados para três classes de consultas segundo a estratégia descendente de avaliação. Na consulta (a), a coleção alvo contém 100 objetos sendo acessados com um predicado simples. Nas outras duas a coleção alvo possui 1000 objetos, sendo que na consulta (b), o predicado é sobre um atributo local à própria coleção, com fator de seletividade igual a 0,1% e na outra consulta (c), o predicado é sobre um atributo que referencia outra coleção que possui 100 objetos, com fator de seletividade igual a 10%.

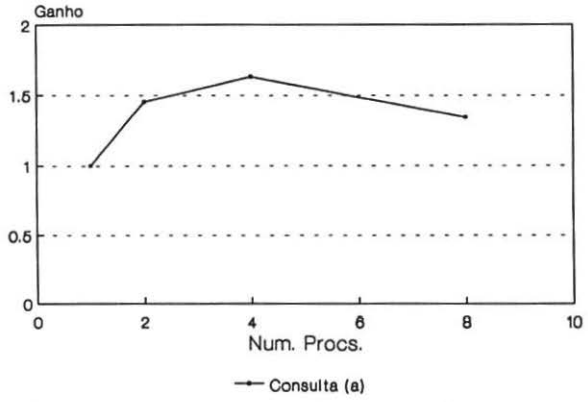

Figura 3 - Curva de ganho para a consulta (a) com tempo de processamento para o processador 1 = 0"645

Na consulta (a) o predicado possui um nível de atributos operando. Apesar da coleção ter sido distribuída totalmente entre os processadores, observa-se que o ganho para 8 processadores é inferior ao ganho de 4 processadores, evidenciando que o pequeno número de objetos era mais adequado à distribuição parcial. Neste caso, o Gerente de Distribuição poderia alocar somente 4 processadores para a operação deixando os demais livres para outras operações.

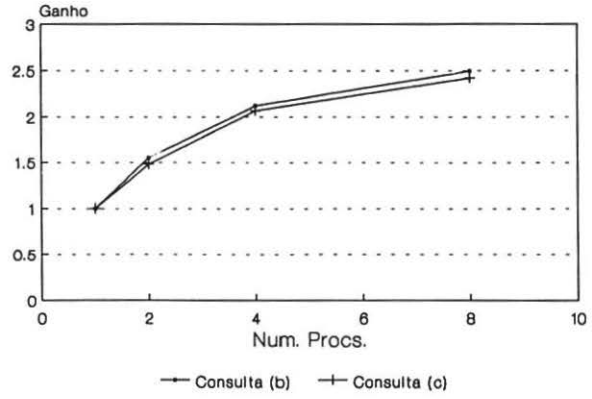

Figura 4 - Curvas de ganho para as consultas (b) e (c) com tempo de processamento para o processador 1 (b) = 6"336 e para o processador 1 (c) = 9"012

Observa-se que nas duas consultas sobre 1000 objetos a curva de ganho é praticamente a mesma, mostrando que embora uma das consultas tenha que acessar objetos de outra coleção concorrentemente com os outros processadores, o ganho foi mantido. O que aumentou foi o tempo de execução isolado conforme já era esperado. A manutenção da taxa de ganho evidencia que características típicas da orientação a objetos não prejudicaram o potencial de ganho do paralelismo.

É importante comentar que embora as medidas de desempenho tenham apresentado ganho em todas as situações, os resultados obtidos foram aquém do esperado. O fato do PARGOA ter utilizado um computador hospedeiro para o NCP I de capacidade inferior àquela utilizada nos experimentos do PARBASE, fez com que os testes apresentados em [Matt91a] fossem repetidos na configuração atual no sentido de avaliar o impacto das novas condições do hardware. De fato, obteve-se uma queda no desempenho e no ganho do processamento das consultas do PARBASE, em relação aos resultados obtidos com a primeira configuração.

Tal fato indica que embora a configuração do NCP I não tenha alterado, a mudança no computador hospedeiro, principalmente na configuração do disco rígido com características inferiores tanto na controladora quanto na capacidade de armazenamento, prejudicaram o potencial de desempenho oferecido tanto pelos algoritmos como pelos processadores do modelo T8 do NCP I.

Embora tenha sido utilizado um PC como computador hospedeiro, o NCP I está sendo integrado com a rede de Sun's do Programa de Engenharia de Sistemas e Computação da COPPE, onde uma estação Sun servirá de hospedeira do NCP I colocando em prática a arquitetura proposta para o GEOTABA da Figura 1.

5 CONSIDERAÇÕES FINAIS

A difusão dos SGBDOOs para aplicações não convencionais, vem sendo realizada com relativo sucesso, através de protótipos experimentais e de alguns produtos que começam a aparecer no mercado. A aceitação desses SGBDOOs para algumas dessas aplicações potenciais, entretanto, está condicionada ao aprimoramento de arquiteturas e de algoritmos para a gerência de objetos para que os produtos ofereçam um desempenho satisfatório.

Foram avaliadas as dificuldades de explorar paralelismo na gerência de objetos. O servidor paralelo de objetos desenvolvido, PARGOA, apresenta soluções para conciliar a distribuição dos objetos de uma coleção, adequado ao processamento paralelo, com o requisito de agrupamento para o processamento de objetos em conjunto. A solução adotada, tira proveito do modelo de memória de disco compartilhado, utilizado na máquina paralela NCP I, que não obriga que os objetos estejam distribuídos fisicamente entre os processadores. Assim, o armazenamento dos objetos no PARGOA não compromete o agrupamento dos objetos nas páginas do disco.

O PARGOA utiliza um algoritmo de gerência de paralelismo para as operações implementadas no servidor de objetos e algoritmos paralelos específicos para a avaliação de predicados. A partir da aferição de tempos de resposta para diversas consultas sobre o modelo de dados orientado a objetos, foi evidenciado o ganho obtido no desempenho do servidor paralelo de objetos. Os resultados alcançados com o PARGOA, confirmam a viabilidade do uso de paralelismo em SGBDs orientados a objetos.

Com os resultados obtidos, observou-se que o desempenho de um SGBDOO pode realmente aumentar, através do uso de um servidor de objetos paralelo em sua arquitetura, o que atinge a um dos objetivos do desenvolvimento deste trabalho. O uso de um servidor paralelo de objetos é ainda mais incentivado, dada a disponibilidade crescente de máquinas paralelas, como

através do uso de redes de estações de trabalho. É importante que novas técnicas sejam estudadas para tirar proveito do potencial desta nova realidade do equipamento, onde os usuários dos SGBDOOs podem ter muito a ganhar.

A curto prazo, tenciona-se repetir os testes realizados com o PARGOA, na nova arquitetura do NCP I através do novo disco adquirido, atualmente em fase de testes na arquitetura piloto. Com a nova realidade de espaço de armazenamento da memória secundária, pretende-se aumentar a base de testes e utilizar 'benchmarks' já definidos para a orientação a objetos [Catt92].

6 REFERÊNCIAS BIBLIOGRÁFICAS

[Agha86] Agha,G. "Actors: A Model of Concurrent Computation in Distributed Systems" MIT Press, 1986.

[Amer91] America,P. "Programmer's Guide for POOL2" POOL2/PTC Distribution Package, Philips Research Laboratories, University of Amsterdam, janeiro 1991.

[Amor91] Amorim,C.L. Citro,R. Souza,A. Chaves,E. "The NCP I Parallel Computer System" Relatório Técnico ES-241/1991, COPPE-Sistemas/UFRJ, abril 1991.

[Arti90] Artificial Intelligence Limited "STRAND^{88} User Manual" Buckingham Release, junho 1990.

[Berg91] Bergsten,B. Couprie,M. Valduriez,P. "Prototyping DBS3, a Shared Memory Parallel Database System" Proceedings First International Conference on Parallel and Distributed Information Systems, Miami, dezembro 1991, pp.226-234.

[Bhid88] Bhide,A. "An Analysis of Three Transaction Processing Architectures" Proceedings of the 14th Int. Conference on Very Large Data Bases, Los Angeles, 1988, pp. 339-350.

[Bora88] Boral,H. "Parallelism and Data Managgement" Proceedings of the 3rd International Conference on Data and Knowledge Bases, Jerusalem, Israel, junho 1988, pp.362-373.

[Catt92] Cattell,R.G.G. Skeen,J. "Object Operations Benchmark" ACM Transactions on Database Systems, v.17(1), março 1992, pp. 1-31.

[DeWi90a] DeWitt,D. Gray,J. "Parallel Database Systems: The Future of Database Processing or a Passing Fad ?" SIGMOD Record v.19 (4), dezembro 1990, pp. 104-112.

[DeWi90b] DeWitt,D. Maier,D. Futtersack,P. Velez,F. "A Study of Three Alternative Workstation-Server Architectures for Object-Oriented Database Systems" Proceedings of the 16th International Conference on Very Large Data Bases, Brisbane, Austrália, agosto, 1990, pp. 107-121.

[DeWi92] DeWitt,D. Gray,J. "Parallel Database Systems: The Future of High Performance Database Systems" Communications of the ACM v.35 (6), junho 1992, pp. 85-98.

[Gard90] Gardarin,G. Valduriez,P. "ESQL: An Extended SQL with Object and Deductive Capabilities, INRIA Research Repport 1185, março 1990.

[Grae90] Graefe,G. "Encapsulation of Paralelism in the Volcano Query Processing System" Proceedings ACM SIGMOD International Conference on Management of Data, Atlantic City, EUA, maio 1990, pp.102-111.

[Hong91] Hong,W. Stonebraker,M. "Optimization of Parallel Query Execution Plans in XPRS" Proceedings First International Conference on Parallel and Distributed Information Systems, Miami, dezembro 1991, pp.218-225.

[Hong92] Hong,W. "Exploiting Inter-Operation Parallelism in XPRS" Proceedings ACM SIGMOD Int. Conf. on Management of Data, San Diego, EUA, junho 1992, pp.19-28.

[Jézé92] Jézéquel,J.M. "Parallelisme Massif et Langage a Objets: Une Approche SPMD", Relatório Técnico INRIA n.1607, fevereiro 1992.

[Laks89] Lakshmi,M.S. Yu,P.S. "Analysis of parallel processing architectures for database systems", Proceedings 1989 International Conference on Parallel Processing, vol I, 1989, pp.83-90.

[Matt91a] Mattoso, M.L.Q. Amorim, C.L. "Uma experiência na implementação de operadores da álgebra relacional no computador paralelo NCP I" Anais VI Simpósio Brasileiro de Banco de Dados, Manaus, maio 1991.

[Matt91b] Mattoso, M.L.Q. "Bancos de Dados e Paralelismo: uma experiência prática." Submetido para apreciação em julho 1991 e aceito para publicação na Revista de Informática Teórica e Aplicada em dezembro 1991.

[Matt93] Mattoso, M.L.Q. "Aspectos de Paralelismo na Gerência de Dados e Objetos no GEOTABA" Dissertação de Tese de Doutorado, Programa de Engenharia de Sistemas e Computação, COPPE/UFRJ, abril 1993.

[Mits92] Mitschang,B. "PRIMA - A Testbed for Database Processing" Anais VII Simpósio Brasileiro de Banco de Dados, Porto Alegre, maio 1992, pp. 21-38.

[Ozsu91] Ozsu,M. Valduriez,P. "Principles of Distributed Systems", Prentice-Hall, 1991.

[Sala91] Salamet,P.B. Chachaty,C. Dageville,B. "Compiling Control into Database Queries for Parallel Execution Management" Proceedings First International Conference on Parallel and Distributed Information Systems, Miami, dezembro 1991, pp.271-279.

[Ston86] Stonebraker,M. "The Case for Shared Nothing" IEEE Database Engineering, v.9(1), março 1986.

[Ston88] Stonebraker,M. "The Design of XPRS" Proceedings of the 14th International Conference on Very Large Data Bases, Los Angeles, 1988, pp. 339-350.

[Ston90] Stonebraker, M., "Third Generation Database System Manifesto", *Proceedings of the 1990 ACM SIGMOD International Conference on Management of Data*, Atlantic City, NJ, maio 1990.

[Vald90] Valduriez,P. "Query Processing in the EDS Parallel Database System", 5. Simpósio Brasileiro de Banco de Dados, Rio de Janeiro, abril 1990, pp. 2-14.

[Wyat92] Wyatt, B.B. Kavi, K. Hufnagel, S. "Paralelism in object-oriented languages: a survey" IEEE Software, v.9(11), novembro 1992, pp.56-66.