Revista UNIMAR 19(4):997-1016, 1997.

ANÁLISE COMPARATIVA DOS PRINCIPAIS *BENCHMARKS* VOLTADOS À ANÁLISE DE DESEMPENHO DE SGBDOOs

Sandro Danilo Gatti*, Ricardo Rodrigues Ciferri* e Carlos José Maria Olguín*

RESUMO. Este artigo apresenta um estudo sobre a técnica de *benchmark* aplicada no contexto de análise de desempenho e efetua uma análise comparativa entre os principais *benchmarks* voltados à análise de desempenho de SGBDOOs, a saber: OO1 Benchmark, HyperModel Benchmark e OO7 Benchmark. Estes são avaliados quanto ao conjunto de transações, à caracterização dos dados que compõem o banco de dados, ao modelo de execução, à facilidade de implementação e à sua representatividade para aplicações orientadas a objetos.

Palavras-chave: banco de dados, orientação a objetos, análise de desempenho, *Benchmark.*

COMPARATIVE ANALYSIS OF THE MAIN BENCHMARKS FOR ODBMS PERFORMANCE EVALUATION

ABSTRACT. This study deals with the benchmark technique for performance evaluation and makes a comparative analysis between the main benchmarks, OO1 Benchmark, HyperModel Benchmark and OO7 Benchmark, used in the context of ODBMS. These benchmarks are evaluated as for the set of transactions, database characterization, execution model, implementation facility and object-oriented application relevance.

Key words: databases, object-orientation, performance evaluation, benchmark.

---

* Departamento de Informática, Universidade Estadual de Maringá, Av. Colombo, 5790, Câmpus Universitário, 87020-900, Maringá-Paraná, Brasil. E-mail: {sdgatti, rrc, olguin}@din.uem.br

Correspondência para Carlos José Maria Olguín.

Data de recebimento: 26/09/97.

Data de aceite: 26/11/97.

INTRODUÇÃO

Sistemas Gerenciadores de Banco de Dados (SGBDs) surgiram com a necessidade de se resolver problemas inerentes a sistemas de processamento de arquivos. Dentre estes problemas, podem-se citar: redundância e inconsistência dos dados, dificuldade no acesso aos dados, anomalias de acesso concorrente, proteção e recuperação de falhas, dentre outros. Um SGBD é composto de dados e de um conjunto de ferramentas para manipular estes dados. Para descrever aspectos relacionados aos dados, um SGBD utiliza um modelo de dados. Os principais tipos de modelos de dados existentes são orientados a registros, isto é, os dados são observados como uma seqüência fixa de campos valorados. Os dados, nestes modelos, assumem uma homogeneidade horizontal e vertical, ou seja, cada tipo de registro possui os mesmos tipos de campos (atributos) e cada campo possui o mesmo tipo de informação em todos os registros. O modelo hierárquico, o modelo de rede e o modelo relacional são exemplos de modelos orientados a registros.

Atualmente, SGBDs relacionais são os mais utilizados comercialmente. Apesar de serem utilizados de forma eficiente em uma enorme gama de aplicações, estes apresentam certas limitações do ponto de vista semântico, além de não atenderem aos requisitos de aplicações não convencionais. Exemplos de aplicações não convencionais são: Sistemas de Informações Geográficas (Ciferri, 1995) e sistemas CAD/CAM. A natureza dos dados armazenados por aplicações não convencionais, a qual focaliza o aspecto espacial das informações, requer que sejam efetuadas modificações em diversos componentes de um SGBD, tais como: processador e otimizador de consultas, método de acesso, gerenciador de transações e armazenamento físico em memória secundária.

Sistemas Gerenciadores de Banco de Dados Orientados a Objetos utilizam os conceitos de persistência, objetos complexos, encapsulamento, hierarquia e polimorfismo, dentre outros, dando ênfase ao aspecto comportamental dos objetos para representar, de uma forma mais apropriada, o mundo real, especialmente para aplicações não convencionais.

Devido ao crescente e recente desenvolvimento de SGBDOOs, diferentes soluções para a implementação deste tipo de sistema têm sido

propostas, tornando imprescindível a utilização de algum mecanismo que possa medir a eficiência destas soluções para auxiliar o direcionamento de futuros trabalhos de pesquisa. Em geral, para o propósito de análise de desempenho, utiliza-se a técnica experimental de *benchmark*. Genericamente, a técnica de *benchmark*, quando aplicada em SGBDs, consiste na execução de um conjunto conhecido de transações ou carga de trabalho, como é comumente denominada, sobre um banco de dados também conhecido (Gray, 1991). Tanto a carga de trabalho quanto o banco de dados podem ser reais ou sintéticos (artificiais). Os resultados gerados por esta técnica são altamente confiáveis, uma vez que o próprio sistema, sendo analisado, é utilizado para a obtenção dos resultados de desempenho.

Este artigo tem por objetivo apresentar uma análise comparativa dos principais *benchmarks* voltados à análise de desempenho de Sistemas Gerenciadores de Banco de Dados Orientados a Objetos: OO1 - Object Operations Benchmark (Cattell, 1992), OO7 Benchmark (Carey *et al*, 1993) e HyperModel Benchmark (Anderson *et al*, 1990). Estes *benchmarks* foram escolhidos para análise por serem os mais utilizados na indústria e no meio acadêmico. Os critérios usados nesta análise são os seguintes: caracterização das transações, caracterização dos dados, geração de dados sintéticos, modelo de execução, nível de abstração, facilidade de implementação, custo $\times$ tempo de implementação, e representatividade para sistemas orientados a objetos.

O restante deste artigo é estruturado da seguinte maneira: a próxima seção apresenta os principais conceitos relacionados à análise de desempenho e à técnica experimental de *benchmark*, destacando as vantagens de sua utilização e descrevendo características sobre a carga de trabalho, dados e medidas de desempenho. A seção 3 apresenta a análise comparativa dos *benchmarks* considerados neste trabalho. As conclusões e extensões são apresentadas na seção 4.

ANÁLISE DE DESEMPENHO

Esta seção apresenta alguns conceitos relacionados com a análise de desempenho, notadamente a técnica de *benchmark*, uma vez que alguns termos aqui apresentados são utilizados para a avaliação dos *benchmarks* considerados neste artigo.

A análise de desempenho ajuda a responder quatro questões (Ciferri, 1995):

- *avaliação de capacidade máxima:* prediz quão eficiente um sistema efetua determinada funcionalidade;
- *comparação entre tecnologias:* ajuda a identificar restrições de desempenho em determinados produtos, indicando reestruturações necessárias, além de possibilitar a identificação de produtos que devem ser melhor pesquisados e melhorados;
- *avaliação de capacidade específica:* verifica se determinada aplicação pode ser executada em determinado sistema computacional com desempenho aceitável;
- *avaliação da relação custo* $\times$ *benefício:* verifica qual é o sistema que se adequa às necessidades de desempenho com o menor custo.

Para cumprir estas questões, a análise de desempenho utiliza-se dos seguintes modelos:

- *modelo analítico:* baseado na obtenção de um conjunto de equações matemáticas, juntamente com algoritmos para resolvê-las, que relacionam medidas de desempenho a parâmetros do sistema sendo analisado. Pelo fato de empregar hipóteses teóricas a respeito do sistema sendo modelado, os resultados gerados tendem a ser imprecisos e seu uso em sistemas reais torna-se reduzido. É mais utilizado durante a fase de projeto, quando se quer ter uma estimativa do seu comportamento;
- *modelo de simulação:* a simulação baseada em computador implica a formalização de um processo, de um sistema ou de qualquer outra coisa do mundo real e na sua posterior implementação em um programa de computador. O modelo de simulação procura reproduzir as atividades dos sistemas sendo analisados de acordo com um conjunto de hipóteses e condições, eliminando a necessidade de experimentação no próprio sistema;
- *modelo experimental:* devido ao fato de utilizar o próprio sistema para a obtenção dos resultados de desempenho, os resultados são altamente confiáveis. Deste modelo fazem parte duas técnicas principais: a técnica de monitoração e a técnica de *benchmark*. A técnica de monitoração consiste em utilizar ferramentas próprias de avaliação estatística presentes no sistema que está sendo analisado. A técnica de *benchmark*

consiste na execução de um conjunto fixo de testes sobre um sistema para avaliar seu desempenho, e, ao contrário da técnica de monitoração, pode ser aplicada na comparação de diferentes sistemas, uma vez que é padronizada, sendo os testes invariáveis e bem definidos.

É importante ressaltar que modelos experimentais dependem da maturidade da tecnologia envolvida. Os modelos analíticos e de simulação podem ser empregados em sistema nascentes, onde ainda não se têm dados operacionais disponíveis. É possível, em alguns casos, utilizar modelos híbridos, que envolvem mais de um dos modelos citados anteriormente. Estes modelos tentam associar as vantagens de cada um dos modelos individuais de forma a obter uma representação do sistema mais realista ou, ainda, procurando facilitar a execução das ferramentas criadas.

Técnica de *benchmark*

Um *benchmark* é composto por uma série representativa de testes funcionais e de desempenho, que são efetuados em um determinado subconjunto de dados, simulando, assim, o ambiente da aplicação alvo (Ciferri, 1995). Os resultados são confiáveis, pois utiliza-se o próprio alvo dos testes para a realização dos mesmos. Entretanto, esta técnica exige uma certa maturidade da tecnologia a ser testada, além de ser uma técnica deveras dispendiosa para a implementação. Uma vez que a tecnologia orientada a objetos já atingiu um grau de maturidade aceitável, é possível a implementação de *benchmarks* voltados à análise de desempenho de SGBDOOs. Para sistemas que possuam (como um de seus principais componentes) um SGBD, utiliza-se a técnica de *benchmark* de banco de dados, uma especialização da técnica genérica de *benchmark*, a qual é baseada no conceito de transação.

O uso de *benchmarks* não está restrito apenas à análise de desempenho. Pode-se utilizar esta técnica para testes de verificação e de correção de implementação e de conformidade com alguns modelos de SGBDs. Sua utilização em testes de correção de programas ocorre mais comumente durante os estágios iniciais de implementação de um SGBD, tal como ocorreu nas primeiras implementações dos SGBDs Ingres e System R, que utilizam o modelo relacional.

Benchmarks de domínio específico

Benchmarks de domínio específico são uma resposta à diversidade de sistemas computacionais em uso (Gray, 1991). Cada *benchmark* deste tipo especifica uma carga de trabalho sintética, caracterizando aplicações típicas naquele domínio de problema. O desempenho desta carga de trabalho, em vários sistemas computacionais, dá, então, uma estimativa do desempenho naquele tipo de problema.

Entretanto, um *benchmark* de domínio específico deve satisfazer quatro critérios:

- *relevante:* deve medir o desempenho de pico e preço/desempenho de sistemas, quando realizando transações típicas dentro daquele domínio;
- *portável:* deve ser fácil de implementar em diferentes sistemas e arquiteturas;
- *escalável:* deve ser aplicável tanto em computadores de pequeno quanto em grande porte, até mesmo em sistemas com processamento paralelo;
- *simples:* deve ser fácil de entender.

Desenvolver um *benchmark* não é uma tarefa fácil. Chaudhri, 1994, citando Stonebraker, diz que qualquer um que desenvolva um *benchmark* está em uma situação em que não há vitória, será apenas criticado: observadores externos encontrarão falhas no *benchmark*, dizendo ser artificial ou incompleto, de uma forma ou outra, enquanto desenvolvedores que não tenham obtido resultados de desempenho satisfatórios o criticarão imerecidamente. Por outro lado, desenvolvedores que obtiveram bons resultados dirão que o *benchmark* é pobre, mas que se deve utilizá-lo de qualquer forma.

Carga de trabalho

A eficiência de um sistema computacional está diretamente relacionado aos fatores que degradam seu desempenho. Esta degradação é provocada pela sobrecarga de execução dos processos ativos, limitando a velocidade na qual estas tarefas são executadas. Cada um destes processos ativos gera uma sobrecarga de execução específica, a qual constitui-se da quantidade de recursos computacionais utilizados e do conseqüente tempo gasto durante o uso destes recursos.

A avaliação de desempenho através da técnica de *benchmark* implica a implementação e posterior execução de um conjunto de programas em

um dado sistema computacional. Ao se avaliar o desempenho, insere-se uma sobrecarga de execução específica, gerada pelo próprio *benchmark*. A esta sobrecarga de execução dá-se o nome de carga de trabalho do *benchmark* e engloba todos os processos criados ou acionados para execução, direta ou indiretamente pelo *benchmark*.

Adicionalmente à carga de trabalho do *benchmark*, podem-se ter processos independentes, os quais geram uma sobrecarga de execução particular, chamada carga de trabalho externa ao *benchmark*. Estes processos, obviamente, também utilizam recursos computacionais e, portanto, também degradam o ambiente. Podemos, desta forma, dividir os processos ativos no sistema em dois grupos: os que pertencem à carga de trabalho do *benchmark* e os processos independentes. A sobrecarga de execução gerada por estes dois grupos caracteriza o que chamamos de carga de trabalho total do sistema computacional.

Tanto os processos criados pelo *benchmark* quanto os processos acionados por ele, os quais são provenientes dos *softwares* aplicativos e do sistema operacional, fazem parte da carga de trabalho do *benchmark*. Em adição, vários processos independentes compartilham os recursos computacionais com os processos pertencentes à carga de trabalho do *benchmark*, influindo diretamente na degradação do desempenho do sistema computacional.

Um aspecto muito importante a ser identificado é o sistema de análise. O sistema de análise é formado por um subconjunto da carga de trabalho, presente no sistema computacional, durante a realização dos testes de desempenho, no que tange ao *software* e, opcionalmente, pelos recursos computacionais presentes para suportar essa carga de trabalho, caso o *hardware* seja objeto de análise. Pode englobar o sistema computacional inteiro ou apenas parte deste.

A identificação do sistema de análise é essencial, uma vez que cada nível de *software* de um sistema computacional multinível exerce influência específica no desempenho, através de sobrecargas de execução distintas. Na Figura 1, apresentamos este panorama, ou seja, um sistema computacional multinível, onde podemos observar a carga de trabalho do *benchmark* (acionada direta ou indiretamente), os processos independentes e o sistema de análise.

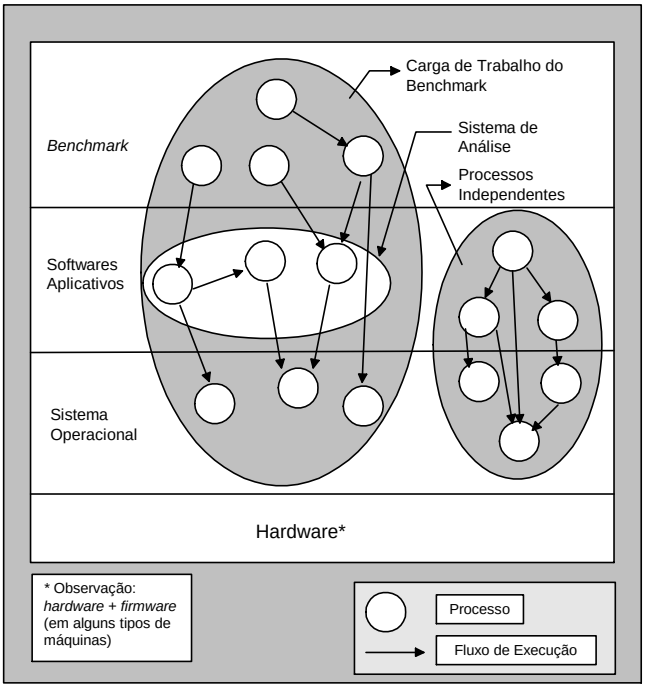

Figura 1. Carga de Trabalho e Sistema de Análise.

Dados em um *benchmark*

Cargas de trabalho distintas influenciam de maneiras distintas o desempenho de um sistema computacional. Da mesma forma, composições de dados distintas degradam o ambiente computacional diferentemente. Assim, a caracterização dos dados de um *benchmark* deve ser feita de maneira não ambígua, ou seja, deve inserir uma constante de degradação única a todos os sistemas a serem analisados.

Há dois modos de se caracterizar dados em um *benchmark.* Através de:

- *dados sintéticos:* são dados gerados artificialmente. Possibilitam a inserção de propriedades especiais de modo a explorar deficiências encontradas ou pressentidas nos sistemas a serem analisados. A geração de dados sintéticos pode optar por uma distribuição uniforme ou não uniforme dos dados. Em geral, a distribuição uniforme é efetuada a partir de uma função aleatória

uniforme, sendo a freqüência para cada valor distinto constante ao longo do tempo. Para obter distribuições assimétricas é utilizada a lei de Zipf (Ciferri, 1995).

- *dados reais:* são um conjunto de dados existentes em uma aplicação referente ao contexto sendo analisado. Esta abordagem não consegue gerar configurações capazes de testar gargalos do sistema, bem como é difícil de se obter um critério controlado para seletividade. Nesta caracterização, é possível se obterem resultados de desempenho condizentes a aplicações desse tipo, desde que os dados sejam oriundos de um aplicação real representativa. Uma outra vantagem é que a descrição destes dados é mais simples, uma vez que implicitamente se relacionam através dos diversos arquivos do banco de dados.

Tipos de medidas

As medidas a serem utilizadas também são fatores importantes. Em geral, são fornecidos três tipos de medidas: medidas de produtividade, medidas de utilização e medidas de tempo de resposta de transações individuais:

- *medidas de produtividade:* consistem no número de transações que são efetuadas em um determinado período de tempo. Também chamadas de *throughput*. Unidades utilizadas: TPS (transações por segundo) e TPM (transações por minuto);
- *medidas de utilização:* servem para detectar possíveis fontes de gargalos. Podem ser reportadas através de quantificação absoluta, dada em medidas de tempo ou capacidade, e quantificação relativa (taxa de utilização), dada em percentagem;
- *medidas de tempo de resposta:* são quantificações unitárias da degradação de desempenho proporcionada pelas transações individuais, medidas através de medidas de tempo, como hora, minuto e segundo. Em medidas de tempo de resposta é importante a definição do espaço de medição, que pode ser interno ou externo. Entende-se por espaço de medição externo o tempo decorrido do processamento e da entrada e saída de dados. O espaço de medição interno reporta apenas o tempo de processamento dos dados, sem considerar tempos de entrada e de saída de dados.

As medidas vistas até agora não reportam um item: o custo. A computação final do custo final envolvido na implantação de um sistema é difícil de ser efetuada. Na prática, os custos são reportados aos custos de *hardware* e *software* e à sua manutenção por um período de cinco anos. Mas raramente o custo individual é utilizado como item de comparação. Dessa forma, o custo é disponibilizado na forma de razão entre grandezas, como custo/desempenho ou desempenho/custo. São as chamadas medidas de custo × benefício. Exemplos de medidas são: K$/TPS[1] ou K$/TPM[2].

ANÁLISE DE DESEMPENHO DE SGBDOOs

Praticamente todos os *benchmarks* voltados à análise de desempenho de SGBDOOs são derivados de um destes três *benchmarks*: OO1, HyperModel e OO7. A semelhança entre eles é que os dados são gerados de forma sintética, com distribuição uniforme e as medidas são obtidas através de tempo de resposta de transações individuais. Entretanto, são muito diferentes entre si se considerarmos as abordagens de seus autores em relação à carga de trabalho, ao modelo de dados, dentre outros fatores.

Em sua avaliação, os seguintes fatores foram considerados: caracterização das transações ou carga de trabalho, caracterização e estruturação do banco de dados, geração de dados sintéticos, modelo de execução, nível de abstração, facilidade de implementação e representatividade para sistemas orientados a objetos. Estes fatores foram escolhidos porque, através deles, é possível avaliar sua representatividade em sistemas orientados a objetos quanto ao conjunto de dados e às transações que são executadas sobre estes dados. Tanto os dados quanto as transações devem refletir o mundo real no qual estão inseridas as aplicações orientadas a objetos.

A seguir, são apresentados cada um dos *benchmarks* alvos do estudo, contendo a caracterização da carga de trabalho, do banco de dados. São apresentados, também, os resultados das observações feitas durante a vigência deste trabalho, bem como algumas observações efetuadas por outros autores.

---

1 K$/TPS: milhares de unidades monetárias por transações por segundo.

2 K$/TPM: milhares de unidades monetárias por transações por minuto.

O *benchmark* OO1

O *benchmark* OO1 (Cattell, 1992), devido à sua simplicidade, foi o primeiro *benchmark* padrão com o qual se tentou medir o desempenho de SGBDs para aplicações de engenharia. Seu intento era medir o desempenho através da navegação e atualizações simples, o custo das interações e a presença e efetividade do cacheamento de clientes. A especificação do OO1 é baseada na experiência de seus desenvolvedores com um *benchmark* para transações simples, em banco de dados que haviam implementado e utilizado na *Sun*.

Carga de trabalho e caracterização do banco de dados

Seu banco de dados é muito simples, sendo composto por dois registros, a saber, *part* (*id*, *type*, *x*, *y*, *build*) e *connection* (*from*, *to*, *type*, *length*). Cada *part* é conectada a três outras *parts* escolhidas aleatoriamente. Observe, na Figura 2, o modelo de objetos da base de dados do OO1, segundo a metodologia OMT. O banco de dados pode ser configurado em três tamanhos (*small*, *large*, *huge*), reside em um servidor em uma rede, enquanto a aplicação roda em uma estação de trabalho, de modo que seja consistente com aplicações em redes, comuns em casos de SGBDOOs.

O *benchmark* é rodado por um único usuário. São incluídas três transações, que devem ser rodadas dez vezes, medindo o tempo de respostas para cada execução, para checar a consistência e o comportamento do cacheamento. As transações são: *lookup*, *traversal*, e *insert*.

Análise do OO1

Em relação aos dados, o OO1 possui um modelo de dados muito simples, o que impede a medição de fechamentos transitivos e outras transações de travessia em ponteiros, muito encontradas em aplicações de engenharia. Não possui medidas para testar efeitos de hierarquias de tipos, herança ou relacionamentos complexos. O *benchmark* opera em objetos aleatoriamente selecionados, o que não condiz às aplicações de engenharia, pois estas não operam em objetos aleatórios. Em muitas aplicações de engenharia, objetos relacionados são acessados sucessivamente, com grande freqüência e mais alto grau que objetos aleatórios e disjuntos. Como resultado deste aspecto aleatório do OO1, agrupamentos (*clustering*) semânticos não podem ser medidos. Ainda

devido a sua simplicidade, comportamentos dinâmicos não podem ser modelados, tais como simulação de eventos. O banco de dados é muito pequeno. Tipicamente, os bancos de dados são muito maiores que os utilizados no *benchmark*.

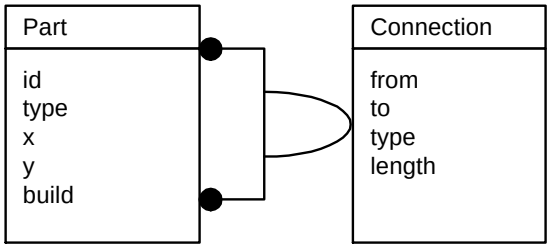

Figura 2. Modelo de Objetos do *Benchmark* OO1.

As transações do OO1 são muito simples, e muitas aplicações de engenharia requerem suporte a transações conceituais de nível mais alto (Carey *et al*, 1993), como, por exemplo, avaliar a eficiência de fechamentos transitivos quando do agrupamento de objetos em torno de um relacionamento, como numa agregação. Não há medida para controle de concorrência e cooperação para avaliar múltiplos usuários editando partes da mesma estrutura de dados. Permite comparações em transações de baixo nível no banco de dados, ao invés de comparações mais significativas no nível da aplicação. O *benchmark* é ótimo para comparações entre sistemas relacionais e sistemas orientados a objetos, mas é fraco para comparações entre sistemas orientados a objetos, tarefa para a qual, de fato, o *benchmark* não foi desenvolvido (Chaudhri, 1994).

O *benchmark* HyperModel

Um outro *benchmark* é o HyperModel (Anderson *et al*, 1990), desenvolvido na Tektronix, Inc. A partir do *benchmark* de transações mais simples da Sun, sobre o qual está baseado o OO1, os desenvolvedores do HyperModel fizeram uma suíte de testes mais ampla para avaliar o desempenho para SGBD de engenharia, baseada em modelo de hipertexto. Se comparado ao OO1, o HyperModel inclui um esquema mais rico e uma gama de transações mais ampla. Na Tektronix, o HyperModel foi implementado sobre dois sistemas: a versão 1 do

VBASE da Ontologic e uma versão para engenharia do GemStone da Servio Logic.

Carga de trabalho e caracterização do banco de dados

Um documento do HyperModel consiste de um número de seções e cada seção é representada por um objeto do tipo *Node*. Há dois subtipos de *Node*: *TextNode* e *FormNode*. Arranjados em forma de árvore, com sete seções (níveis), cada nó com *fan-out* cinco. Os nós internos são do tipo *Node*, enquanto que as folhas podem ser *TextNode* ou *FormNode*. Os *Nodes* são inter-relacionados através de três relacionamentos: *parent/children*, *partOf/parts*, e *refTo/refFrom*. Observe, na Figura 3, o modelo de objetos do HyperModel, segundo a metodologia OMT.

O relacionamento *parent/children* tem cardinalidade 1-N e é utilizado para modelar a estrutura de agregação recursiva das seções dentro de um documento. Além disso, os filhos (*children*) de um dado pai (*parent*) são ordenados. O relacionamento *partOf/parts* é um relacionamento M-N, cuja restrição é ser hierárquico. Isto é, embora *parts* possam compartilhar *subparts*, o relacionamento *partOf/parts* é acíclico. E, finalmente, o relacionamento *refTo/refFrom* é um relacionamento arbitrário M-N, que modela *links* de hipertexto.

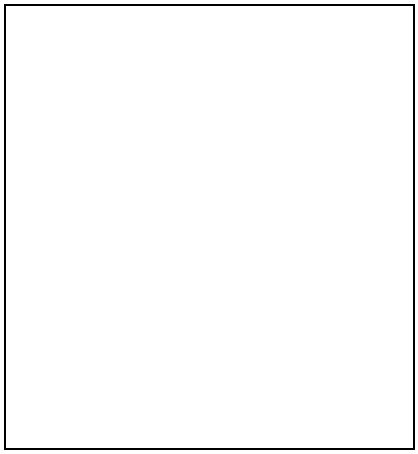

Figura 3. Modelo de Objetos do *Benchmark* HyperModel.

O HyperModel implementa vinte transações, com tempos requeridos como *cold* (cache não inicializado) e *warm*. As transações são de busca

por nome, por faixa, em grupo, por referência, busca seqüencial, fechamento e transações de edição.

Análise do HyperModel

É um *benchmark* mais complexo, especialmente em relação ao conjunto de transações, do que o OO1. Entretanto, ainda não há noção semântica de objetos complexos no *design* do HyperModel. Inclui diversas transações de travessias *read-only*, que parecem ter sido desenvolvidas apenas para serem diferentes umas das outras, e não o proposto, que é de testar o desempenho/*design* de sistemas de banco de dados orientado a objetos (Carey *et al*, 1993). Não há testes desenvolvidos para consultas de objetos, atualizações em objetos indexados versus objetos não-indexados, atualizações repetidas em objetos e várias outras características relacionadas com SGBDOOs. O HyperModel apresenta múltiplos tipos de conexões, mas não define nenhum tipo de medida ou de sugestões de como os dados deveriam ser agrupados de acordo com as várias conexões, sem o que uma estrutura de dados mais complexa deixa de ser útil. Além disso, é difícil de implementar consistentemente a partir das especificações publicadas. Também deveria incluir o tratamento de versões, um conceito importante em aplicações CAD/CAM/CASE.

O *benchmark* OO7

Surgido na Universidade de Wisconsin-Madison, o *benchmark* OO7 (Carey *et al*, 1993) tem a intenção de oferecer um perfil amplo do desempenho de SGBDOOs, além de ser sugestivo em diferentes aplicações CAD/CAM/CASE.

Dentre as características avaliadas pelo OO7, temos:

- velocidade de muitos diferentes tipos de travessias (*traversal*) de ponteiros, incluindo travessias em dados em cache, em dados residentes em disco, travessias esparsas e densas;
- eficiência de diferentes tipos de atualizações, incluindo atualizações a campos de objetos indexados e não-indexados, atualizações repetidas, atualizações esparsas, atualizações em dados em cache, além da criação e remoção de objetos;
- o desempenho do processador de consultas em diferentes tipos de consultas.

Carga de trabalho e caracterização do banco de dados

Uma vez que o *benchmark* OO7 foi desenvolvido para testar diferentes aspectos do desempenho do sistema, a estrutura de seu banco de dados e as transações são não-triviais. Há três tamanhos para o banco de dados do OO7: *small* (pequeno), *medium* (médio), e *large* (grande). Apresentamos, na Figura 4, o modelo de objetos do *benchmark* OO7, segundo a metodologia OMT. Um componente chave do banco de dados do OO7 é um conjunto de *composite parts*. O conjunto de todas as *composite parts* forma o que é chamado de *design library*, dentro do banco de dados do OO7. Cada *composite part* possui alguns atributos, incluindo os atributos inteiros *id* e *buildDate*, e um *array* de caracteres *type*. Associado a cada *composite part* há um objeto *document*, que modela uma pequena quantia de documentação associada com a *composite part*. Cada *document* tem um atributo inteiro *id*, um atributo *title* e uma *string* de caracteres chamada *text*. Uma *composite part* e seu *document* são conectados por uma associação bidirecional.

Cada *composite part* tem um grafo associado de *atomic parts*. Intuitivamente, as *atomic parts*, dentro de uma *composite part*, são unidades das quais uma *composite part* é construída. Uma *atomic part*, em cada grafo, é chamado de *root part*. Cada *atomic part* possui os atributos inteiros *id*, *buildDate*, *x*, *y*, e *docId*, e o *array* de caracteres *type*. Além destes atributos, cada *atomic part* é conectada via uma associação bidirecional a várias (3, 6, 9) outras *atomic parts*. Para assegurar conectividade completa, inicialmente uma conexão é adicionada a cada *atomic part*, de modo a conectá-las em anel; mais conexões são adicionadas, então, aleatoriamente.

As conexões entre *atomic parts* são implementadas interpondo um objeto de conexão entre cada par de partes atômicas conectadas. As conexões possuem dados, contendo o campo inteiro *length* e o *array* de caracteres *type*. Note que a união de todas as partes atômicas corresponde ao grafo do OO1; entretanto, no OO7, este grafo é quebrado em unidades semânticas. Assim, o OO7 oferece uma oportunidade de testar o quão eficiente os vários produtos de SGBDOOs são, ao tratar de objetos complexos.

A *design library*, que contém as *composite parts* e suas *atomic parts* associadas (incluindo os objetos de conexão) e *documents*, é a base do banco de dados do OO7. Entretanto, um conjunto de *composite parts*, por si só, não é suficientemente estruturada para suportar todas as transações

que seriam necessárias incluir no *benchmark*. Assim, foi adicionada a uma *assembly hierarchy* ao banco de dados. Os objetos *assembly* correspondem aos elementos de nível mais alto na aplicação sendo modelada. Por exemplo, em uma aplicação VLSI CAD, um *assembly* corresponderia ao esboço de um arquivo de registro ou a uma ALU. Cada *assembly* é feito ou de *composite parts* (*base assembly*) ou de outros *assemblies* (*complex assembly*).

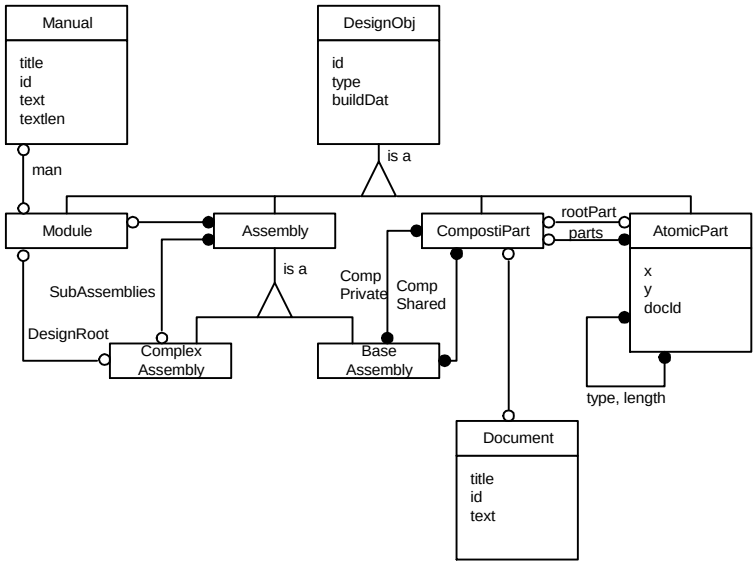

Figura 4. Modelo de Objetos do *Benchmark* OO7.

O primeiro nível da *hierarchy assembly* consiste de objetos *base assembly*. Possuem os atributos inteiros *id* e *buildDate*, e o *array* de caracteres *type*. Cada objeto desta categoria possui uma associação bidirecional com três *composite parts* compartilhas e três não compartilhadas. Níveis mais altos na *hierarchy assembly* são compostos de *complex assemblies*. Cada *complex assembly* tem os atributos inteiros usuais *id* e *buildDate*, e o *array* de caracteres *type*; adicionalmente, tem uma associação com três *subassemblies*, que podem ser *base assemblies*, se estiver no nível dois da *assembly hierarchy*, ou outros *complex assemblies*, se estiver no nível mais alto da hierarquia. Há sete níveis de hierarquia.

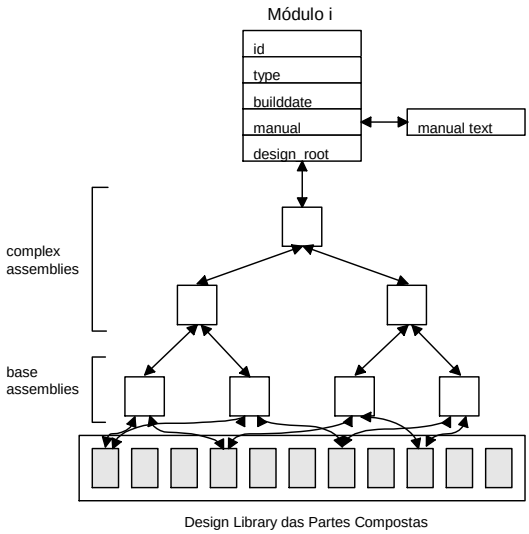

Figura 5. Esquema de um *Module* e sua *Assembly Hierarchy*.

Cada *assembly hierarchy* é chamada de módulo. Módulos devem modelar as maiores sub-unidades do banco de dados da aplicação e são utilizados extensivamente nas cargas de trabalho multiusuário. Módulos possuem os seguintes atributos escalares: os inteiros *id* e *buildDate*, e o *array* de caracteres *type*. Cada módulo possui também um objeto associado *Manual*, que é uma versão maior de um *document*. *Manuals* são incluídos para testar a eficiência em lidar com objetos muito grandes, mas simples. Para melhor compreensão, observe, na Figura 5, um modelo de um módulo, com sua hierarquia *assembly*.

O OO7 inclui transações de travessia (através das quais é medido o desempenho do SGBD na navegação em ponteiros), de consulta (que não modificam o banco de dados) e modificações estruturais de inserção e de remoção. As transações de travessia do OO7 são implementadas como métodos de objetos no banco de dados. Uma travessia navega proceduralmente de objeto em objeto, invocando o método apropriado em cada objeto. Da mesma forma que as travessias, as consultas devem ser executadas *cold* e *hot*, além disso, todas são somente de leitura.

Análise do OO7

Da mesma forma que os *benchmarks* anteriormente citados, o OO7 possui um banco de dados pequeno (500 MB para a configuração *large*), embora seja muito maior que o do OO1 e do HyperModel; aplicações de engenharia atuais atingem tamanhos na faixa dos *gigabytes*. É difícil aumentar o tamanho dos objetos no OO7 sem modificar a descrição das classes. Não há controle sobre o *layout* dos dados (*clustering*) (Tiwary *et al*, 1995). Não é avaliado o impacto que múltiplos usuários impõem ao sistema. O OO7 utiliza-se de *sets*, de *bags*, e de outras estruturas de dados persistentes ao criar seu banco de dados. Embora resulte em uma implementação mais otimizada, isto torna o código não portável. O OO7 possui um suporte à instrumentação muito primitivo (Tiwary *et al*, 1995); as únicas informações disponíveis são o tempo total das transações, divididas em tempo da CPU do sistema e tempo da CPU do usuário; sua implementação não prevê a adição de contadores, rotinas de impressão em tela, etc.

Baseando-se em observações na indústria, chegou-se à conclusão de que o OO7 é inicialmente um *benchmark* para aplicações CAD genéricas. Além disso, não oferece um mapeamento satisfatório para aplicações financeiras, de telecomunicações e multimídia.

CONCLUSÕES E EXTENSÕES

Este artigo apresentou uma análise comparativa dos principais *benchmarks* voltados à análise de desempenho de Sistemas Gerenciadores de Banco de Dados Orientados a Objetos, a saber: OO1 Benchmark, OO7 Benchmark e HyperModel Benchmark. Na avaliação destes *benchmarks*, foram considerados os seguintes aspectos: caracterização das transações ou carga de trabalho, caracterização e estruturação do banco de dados, geração de dados sintéticos, modelo de execução, nível de abstração, facilidade de implementação e representatividade para sistemas orientados a objetos. Adicionalmente, este artigo apresentou os principais conceitos relacionados à análise de desempenho e à técnica experimental de *benchmark*, destacando as vantagens de sua utilização e descrevendo características sobre a carga de trabalho, dados e medidas de desempenho. Através da análise comparativa dos *benchmarks* acima citados, pôde-se verificar que:

- algumas características são comuns, tais como geração de dados sintéticos com distribuição uniforme, utilização do tempo de

resposta de transações individuais como medida de desempenho, carga de trabalho mista e escalabilidade;
- o modelo de dados e a carga de trabalho são orientados a aplicações de engenharia;
- os bancos de dados possuem tamanho reduzido, bem inferior ao tamanho encontrado em aplicações não convencionais reais.

Veja, na Tabela 1, uma síntese dos resultados obtidos na comparação entre os *benchmarks* estudados.

Tabela 1. Tabela comparativa entre os *benchmarks* estudados.

	OO1	**HyperModel**	**OO7**
Geração do Banco de Dados	Sintética e uniforme	Sintética e uniforme	Sintética e uniforme
Aceitação da Indústria	Limitada	Nenhuma	Ampla
Carga de Trabalho	Restrita	Abrangente	Abrangente, diversificada e relevante
Número de Usuários	Mono	Mono	Mono/Multiusuário
Orientação	Engenharia	Engenharia	Engenharia
Medidas de Desempenho	Tempo de Resposta de Consultas Individuais	Tempo de Resposta de Consultas Individuais	Tempo de Resposta de Consultas Individuais
Especificação	Detalhada	Limitada	Detalhada
Tamanho do Sistema	Escalável	Escalável	Escalável
Operações Utilitárias	Nenhuma	Nenhuma	Nenhuma
Código Fonte do *Benchmark*	Morgan Kaufmann	Difícil Encontrar	ftp://ftp.cs.wisc.edu

Baseando-se nos estudos das especificações e implementações dos *benchmarks* alvos deste trabalho e em comparações onde foram considerados os aspectos propostos (carga de trabalho, ao conjunto de transações, modelo de dados, dentre outros), conclui-se que o *benchmark* OO7 apresenta características específicas que o tornam mais representativo, tais como: suporte multiusuário, modelo de dados rico, tratamento de objetos complexos, especificação consistente e não ambígua e carga de trabalho diversificada. A carga de trabalho do OO7 contém diferentes tipos de travessias de ponteiros (isto é, travessias em dados em cache, em dados residentes em disco, travessias esparsas e densas) e diferentes tipos de atualizações (isto é, atualizações a campos de objetos indexados e não-indexados, atualizações repetidas, atualizações esparsas e atualizações em dados em cache), além da criação e da remoção de objetos.

Outra conclusão a que se pôde chegar em relação a este estudo é que, apesar da evolução dos *benchmarks* voltados à análise de desempenho de SGBDOOs, algumas características importantes estão ausentes, como:

suporte a múltiplos bancos de dados e/ou bancos de dados distribuídos, incorporação de medidas de produtividade, utilização, custo × benefício, e gerenciamento de grandes quantidades de dados.

Dentre as possíveis extensões aos *benchmarks* estudados, pode-se citar:

- extensão do *benchmark* OO7 para o suporte de Sistemas de Informações Geográficas, através da incorporação de transações que tratem o aspecto georeferenciado das informações;
- criação de um mecanismo genérico para geração de dados sintéticos que permita seletividade controlada de grandes quantidades de dados;
- incorporação de testes que analisem a eficiência no armazenamento terciário, característica típica de aplicações não convencionais com *petabytes* de dados armazenados em dispositivos de armazenamento terciário (*tape silos* e *compact-disk juke boxes*);
- adição de novos *benchmarks* voltados à análise de desempenho de SGBDOOs, especialmente fora do âmbito da engenharia, como áreas financeiras, multimídia, dentre outras.

REFERÊNCIAS BIBLIOGRÁFICAS

ANDERSON, T.L., BERRE, A.J., MALLISON, M., PORTER III, H.H. & SCHNEIDER, B. *The hypermodel benchmark*. In: Lecture notes in computer science, vol. 416, p.317-331. Springer Verlag, 1990.

CATTELL, R.G.G. & SKEEN, J. Object operations benchmark. *ACM Transactions on Database Systems*, *17*(1):1-31, 1992.

CAREY, M.J., DEWITT, D.J. & NAUGHTON, J.F. The OO7 Benchmark, *Proceedings of ACM SIGMOD Conference*, Washington DC, 1993, 12-21.

CHAUDHRI, A.B. & REVELL, N. Benchmarking object databases: past, present and future. *Object-Oriented Databases*, London, 1994

CIFERRI, R.C. *Um Benchmark voltado para análise de desempenho de sistemas de informações geográficas*. Campinas, 1995. Dissertação (Mestrado em Ciência da Computação) - Universidade Estadual de Campinas.

GRAY, J. *The benchmark handbook for database and transaction processing systems*. Morgan Kaufmann Publishers, 1991.

TIWARY, A., NARASAYYA, V. & LEVY, H., Evaluation of OO7 as a system and application benchmark. *OOPSLA Conference*, Los Angeles, 1995.